UD 3: Bases de datos y servicios REST
1.- Introducción

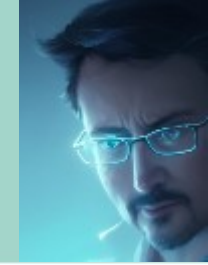
Desarrollo Web en Entorno Servidor - joseramon.profesor@gmail.com

Objetivos de la sesión:

- *Entender las **ventajas** de los proyectos realizados con **Spring Boot** .*
- *Aprender a **crear un proyecto de cero** (HolaMundo) con Spring Boot.*
- *Comprender las **equivalencias de configuración entre Spring MVC y Spring Boot**.*
- *Saber **arrancar nuestra webapp desde fuera del Eclipse** por linea de comandos sin necesidad de un servidor Tomcat.*

UD 3: Bases de datos y servicios REST
1.- Introducción

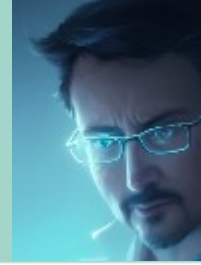
Desarrollo Web en Entorno Servidor - joseramon.profesor@gmail.com



Ya sabemos realizar aplicaciones con el Framework Spring utilizando Spring MVC, pero ….

¿ Por que motivo nos vamos a pasar a Spring Boot ?

¿ Que ventajas nos puede aportar?



Spring Framework tiene una gran potencia, pero puede ser bastante costoso realizar la configuración inicial y la preparación de las aplicaciones para producción.

Spring Boot nos **simplifica la tarea de configuración y el proceso de la generación final del ejecutable** gracias a sus 2 principales mecanismos:

· **Contenedor de aplicaciones integrado:**

Esto nos permite ejecutar nuestra aplicaciones web como un archivo ".jar". Esto significa que ya *no es obligatorio subir un fichero ".war" al Tomcat*. Spring Boot puede tener el Tomcat integrado y ejecutarse de manera independiente. Esto puede ser muy útil en una arquitectura de microservicios con servicios REST. Ojo, porque cuando tenemos un modelo MVC basado en JSP puede dar problemas sin una configuración adecuada.

· **Starters:**

Spring Boot nos ofrece una serie de dependencias, llamadas starters, que podemos añadir a nuestro proyecto dependiendo de lo que necesitemos: acceder a una base de datos, crear un servicio REST, conectar con Apache ActiveMQ,...

Por ejemplo, con la dependencia de Spring Boot Starter Actuator se incorporan las dependencias necesarias para empezar a generar métricas tanto de la JVM como de la aplicación en sí (tiempos de respuesta, errores,...)

UD 3: Bases de datos y servicios REST
1.- Introducción

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

Aunque no es obligatorio utilizar Spring Boot para poder proporcionar acceso a datos a una aplicación hecha con el framework Spring MVC, en la práctica se suele utilizar Spring Boot gracias a sus ventajas, por lo que **nuestro primer paso será crear nuestra primera aplicación en Spring Boot con un HolaMundo y a cotinuación migrar la aplicación realizada en la UD anterior a Spring Boot.**

¿Que pasos realizaremos para migrar nuestra aplicación de Spring MVC a Spring Boot?

Los **pasos a realizar para migrar nuestra aplicación a Spring Boot** son:

1º Crear un proyecto nuevo Spring Boot con las dependencias que utilizaremos en esta UD para acceder a Bds

2º Configurar este proyecto para que inicialmente nos muestre la página HolaMundo

3º Configurar el proyecto para incorporar las clases (ficheros Java) y las vistas(ficheros JSP) del proyecto realizado en la UD anterior

1º Crear un proyecto nuevo Spring Boot :

Para crear un proyecto Spring Boot hay **2 formas** sencillas:

1º forma: Utilizar el IDE “STS”, que es una versión “hecha a medida” por Spring utilizando como base Eclipse.

2º forma: Conectarse a la página web de Spring (**https://start.spring.io)** para generar el proyecto maven con las dependencias de Spring Boot y luego importar el proyecto.

A continuación optaremos por esta **segunda forma** porqué no queremos depender de ningún IDE.

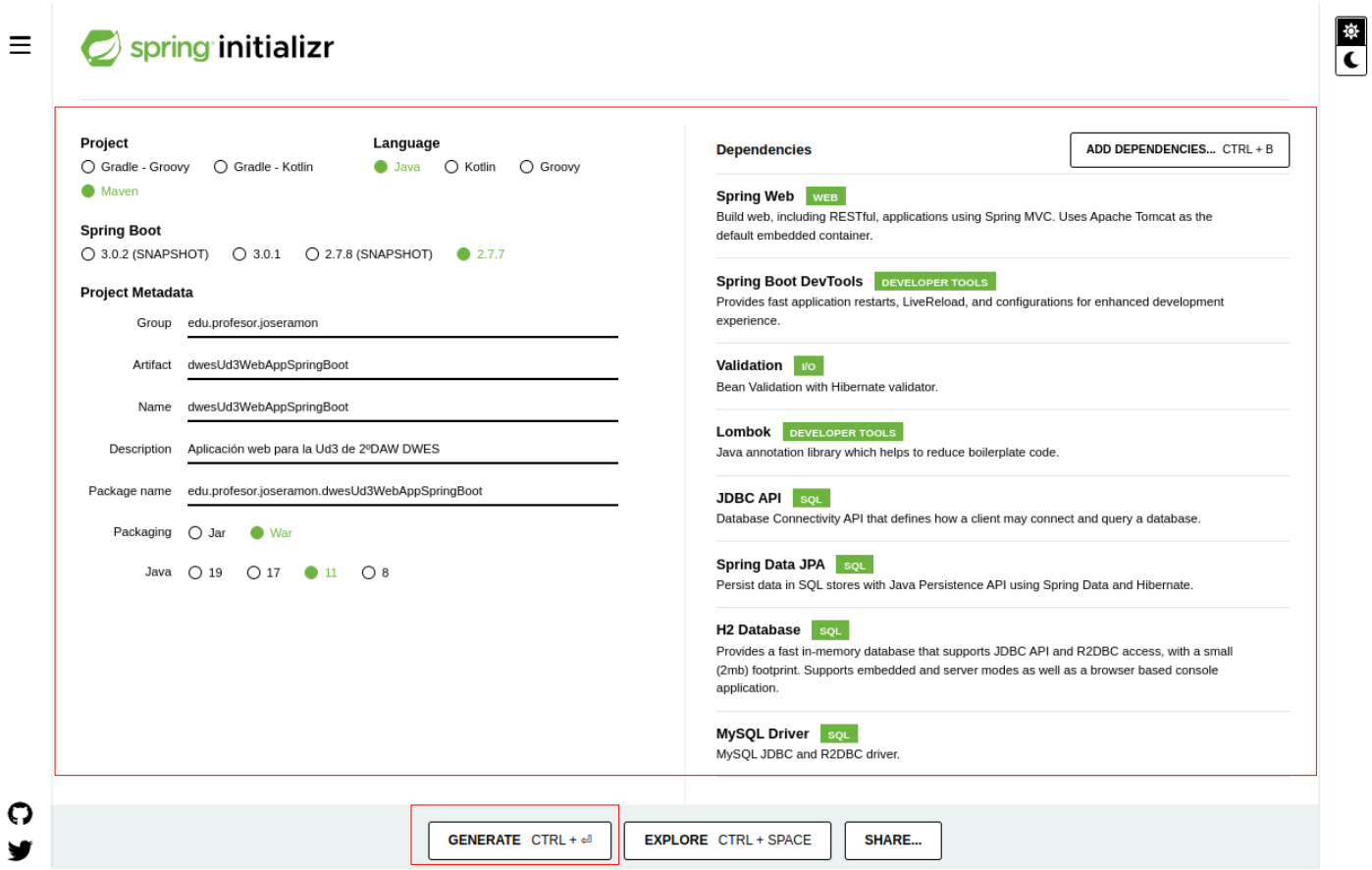
UD 3: Bases de datos y servicios REST
1.- Introducción

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

… continuación 1º Crear un proyecto nuevo Spring Boot :

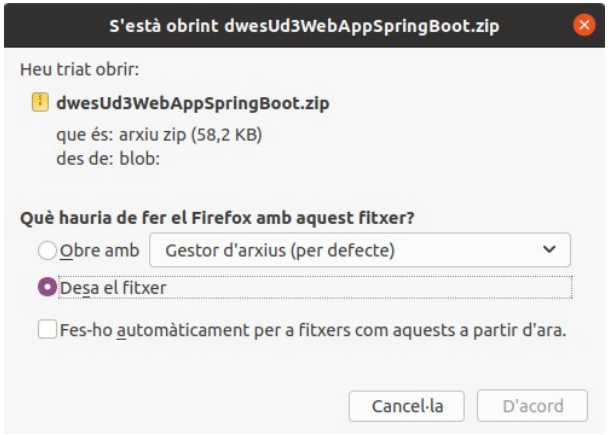
Acceder a “start.spring.io” e introducir los siguientes datos (Project Metadata y Dependencias) teniendo en cuenta que hay que cambiar el group por **“edu.alumno.nombreAlumno”.** Por ejemplo: “edu.alumno.jose”.

Para generar el proyecto se debe pulsar en **“Generate CTRL + Intro”**.

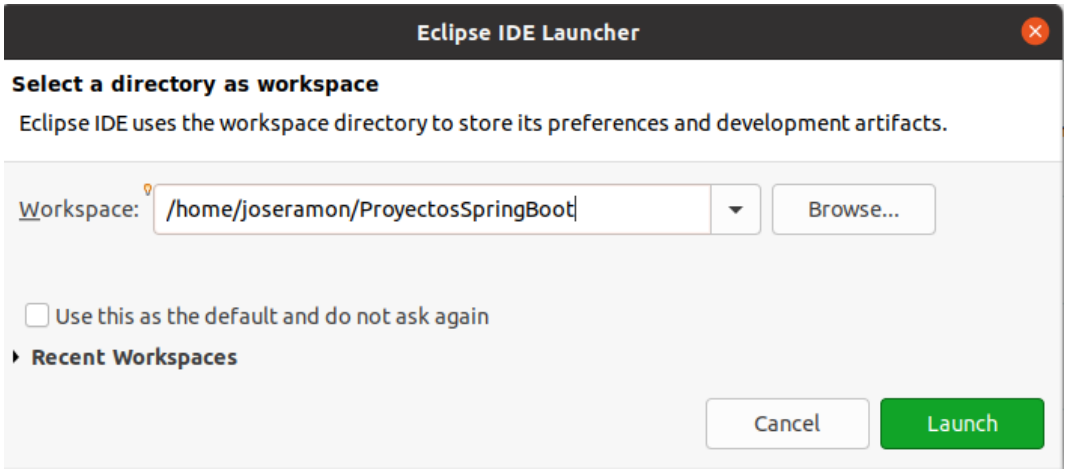


... continuación 1º Crear un proyecto nuevo Spring Boot :

Guardamos el zip que contiene el proyecto en vez de abrirlo:



A continuacion crearemos una **carpeta "ProyectoSpringBoot"** donde iremos dejando todos nuestros proyectos SpringBoot y abrimos el Eclipse seleccionando esta nueva carpeta como directorio del nuevo Workspace que dedicaremos a los proyectos de esta UD.



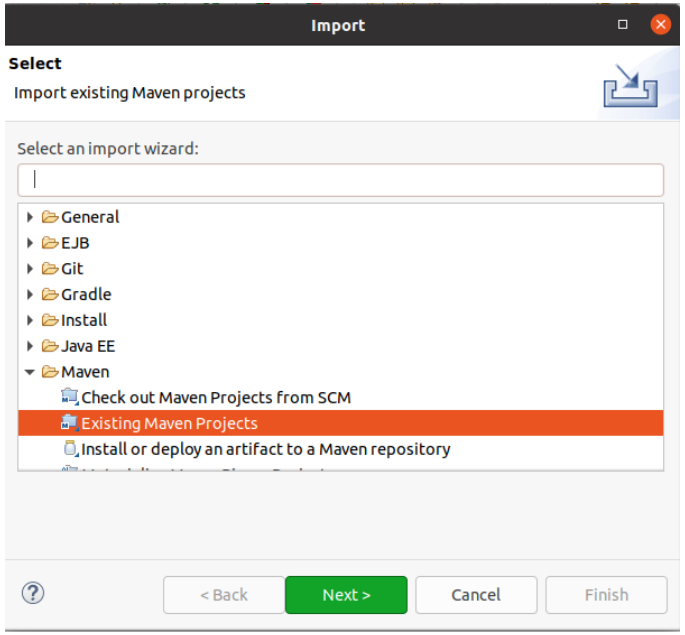
El siguiente paso será extraer el zip con proyecto dentro de esta carpeta

UD 3: Bases de datos y servicios REST
1.- Introducción

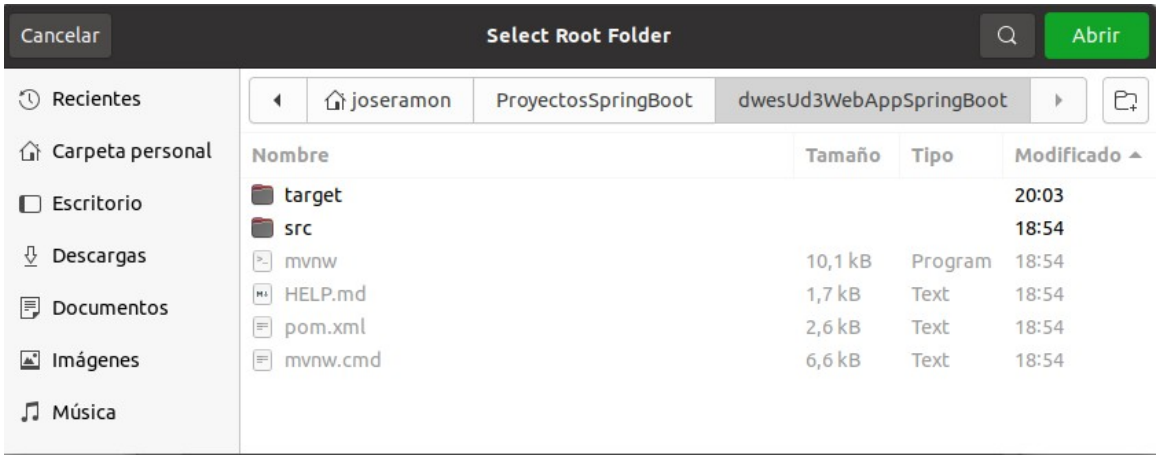
Desarrollo Web en Entorno Servidor - joseramon.profesor@gmail.com

... continuación 1º Crear un proyecto nuevo Spring Boot :

Importamos en Eclipse el nuevo proyecto mediante el menú **"File\Import\Existing Maven Projects"**:

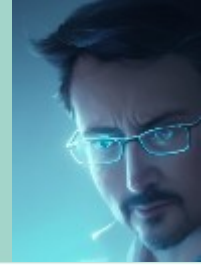


Y deberemos **seleccionar la carpeta** donde hemos descomprimido el proyecto Spring Boot.



UD 3: Bases de datos y servicios REST
1.- Introducción

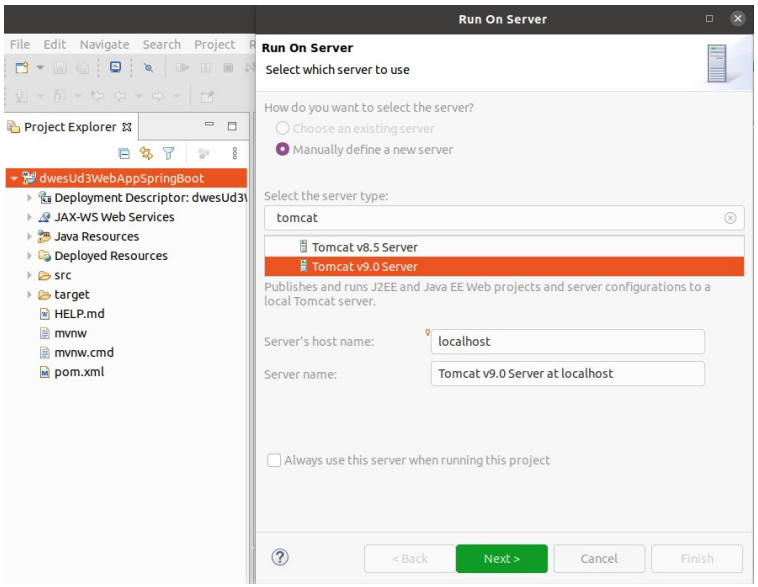
Desarrollo Web en Entorno Servidor - joseramon.profesor@gmail.com

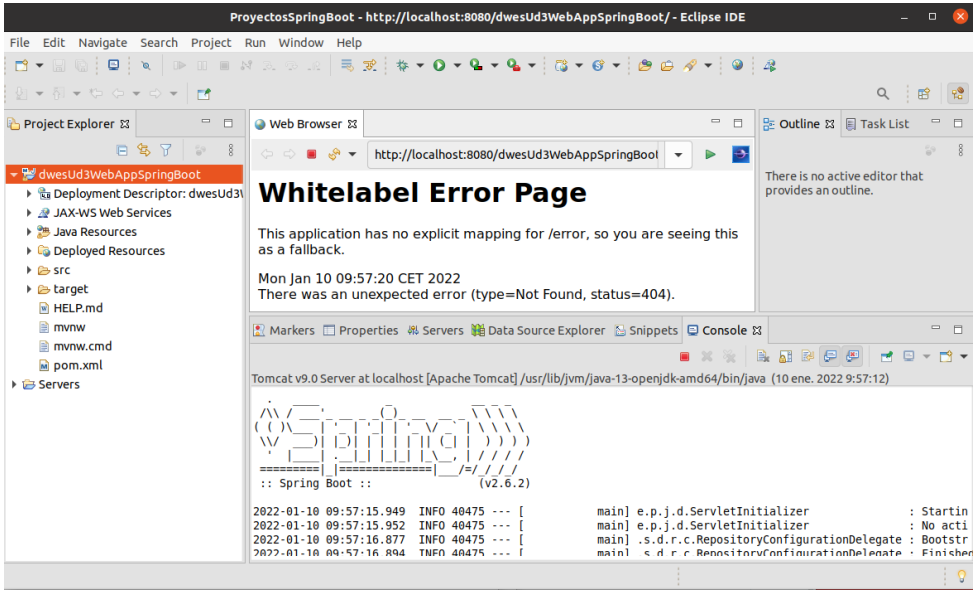


... continuación 1º Crear un proyecto nuevo Spring Boot :

¡Creamos un servidor de Tomcat 9.0 para el nuevo workspace y procedemos a arrancar la app. Puede que tarde un poco porque actualizará librerías de jakarta.

La mala noticia es que al no tener ningún controller fallará al buscar "/" dando un error 404. Adicionalmente tampoco le hemos dicho donde encontrar las paginas web de la vista, ni el hecho de que tendrán extensión ".jsp" :

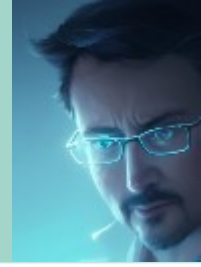




A continuación explicaremos como configurar la aplicación web y nuestro primer controller con su página jsp para hacer nuestro HolaMundo en Spring Boot.

UD 3: Bases de datos y servicios REST
1.- Introducción

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



2º Configuración del proyecto para generar HolaMundo :

Antes de nada revisemos la configuración de nuestro **antiguo proyecto Spring MVC**, y en concreto como teníamos configurado el fichero **"web.xml" donde configurábamos el dispatcher servlet** :

El **DispatcherServlet** era quien implementaba el patrón "Front Controller", quien se encarga de coordinar las peticiones HttpRequest hacia los manejadores (handlers). Esto nos permitía indicar donde estában los controllers.

En nuestro caso en la webApp Spring MVC le decíamos que la configuración se encontraba en el fichero alumno-servlet.xml.

web.xml

```
1  <!--  webapp/WEB-INF/web.xml  -->
2  <web-app xmlns="http://java.sun.com/xml/ns/javaee" xmlns:xsi="http:
3  <display-name>Lista de alumnos</display-name>
4  <welcome-file-list>
5  <welcome-file>/login</welcome-file>
6  </welcome-file-list>
7
8  <servlet>
9          <servlet-name>dispatcher</servlet-name>
10         <servlet-class>
11              org.springframework.web.servlet.DispatcherServlet
12         </servlet-class>
13         <init-param>
14             <param-name>contextConfigLocation</param-name>
15             <param-value>/WEB-INF/alumno-servlet.xml</param-value>
16         </init-param>
17           <load-on-startup>1</load-on-startup>
18 </servlet>
19 <servlet-mapping>
20         <servlet-name>dispatcher</servlet-name>
21         <url-pattern>/</url-pattern>
22 </servlet-mapping>
23 </web-app>
```

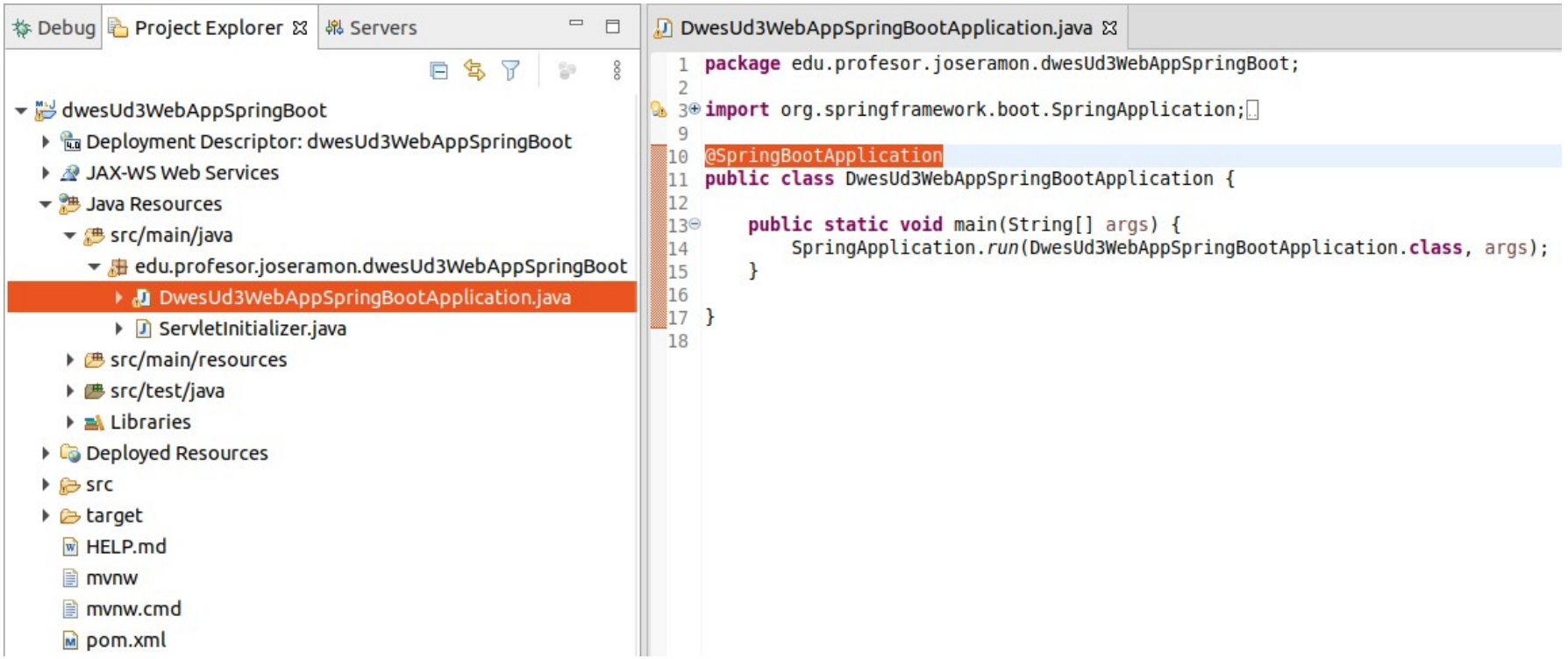
¿Cual es la configuración equivalente en Spring Boot?

UD 3: Bases de datos y servicios REST
1.- Introducción

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

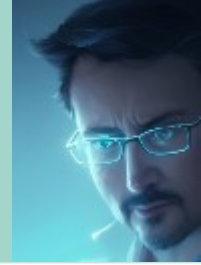
... continuación 2º Configuración del proyecto para generar HolaMundo :

Por defecto, Spring Boot nos ha creado una clase principal DwesUd3WebAppSpringBootApplication con la notación **@SpringBootApplication**. Si quisieramos podríamos configurar una clase que actuara de Dispatcherservlet, pero la notación @SpringBootApplication junto con la la dependencia spring-mvc del fichero pom.xml hace que Spring Boot configure esta clase como la clase principal donde configurar el contexto del DispatcherServlet:



UD 3: Bases de datos y servicios REST
1.- Introducción

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



... continuación 2º Configuración del proyecto para generar HolaMundo :

Por otra parte, en el proyecto de la Ud2 le indicabamos que la configuración del contexto del DispatcherServlet se realizaba desde alumno-servlet.xml:

web.xml

```
<!--  webapp/WEB-INF/web.xml  -->
<web-app xmlns="http://java.sun.com/xml/ns/javaee" xmlns:xsi="http:
<display-name>Lista de alumnos</display-name>
<welcome-file-list>
<welcome-file>/login</welcome-file>
</welcome-file-list>

<servlet>
        <servlet-name>dispatcher</servlet-name>
        <servlet-class>
             org.springframework.web.servlet.DispatcherServlet
        </servlet-class>
        <init-param>
            <param-name>contextConfigLocation</param-name>
            <param-value>/WEB-INF/alumno-servlet.xml</param-value>
        </init-param>
          <load-on-startup>1</load-on-startup>
</servlet>
<servlet-mapping>
        <servlet-name>dispatcher</servlet-name>
        <url-pattern>/</url-pattern>
</servlet-mapping>
</web-app>
```

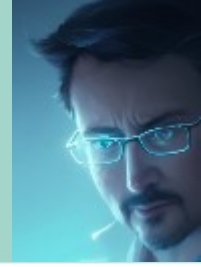
alumno-servlet.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
        xmlns:context="http://www.springframework.org/schema/context"
        xmlns:mvc="http://www.springframework.org/schema/mvc"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.spri
        http://www.springframework.org/schema/mvc http://www.springframework.org/schema
        http://www.springframework.org/schema/context http://www.springframework.org/sc
        <context:component-scan base-package="org.profesor.joseramon" />
        <mvc:annotation-driven />
<bean
        class="org.springframework.web.servlet.view.InternalResourceViewResolver">
        <property name="prefix">
            <value>/WEB-INF/views/</value>
        </property>
        <property name="suffix">
            <value>.jsp</value>
        </property>
</bean>
<bean id="multipartResolver"
        class="org.springframework.web.multipart.commons.CommonsMultipartResolver">
</bean>

<!-- Internacionalización i18n -->
<bean id="messageSource"
    class="org.springframework.context.support.ReloadableResourceBundleMessageSource">
    <property name="basename" value="classpath:messages" />
    <property name="defaultEncoding" value="UTF-8" />
</bean>
<bean id="localeResolver"
        class="org.springframework.web.servlet.i18n.SessionLocaleResolver">
        <property name="defaultLocale" value="es" />
</bean>
<mvc:interceptors>
    <bean id="localeChangeInterceptor"
        class="org.springframework.web.servlet.i18n.LocaleChangeInterceptor">
        <property name="paramName" value="language" />
    </bean>
</mvc:interceptors>
<!-- Fin internacionalización i18n -->

<mvc:resources mapping="/webjars/**" location="/webjars/"/>
<mvc:resources mapping="/imagenes/**" location="/resources/imagenes/"/>
</beans>
```

Sin alumno-servlet.xml, ¿Quien se encarga ahora de realizar dicha configuración? ¿Como le decimos que package escanerar para encontrar los controladores?

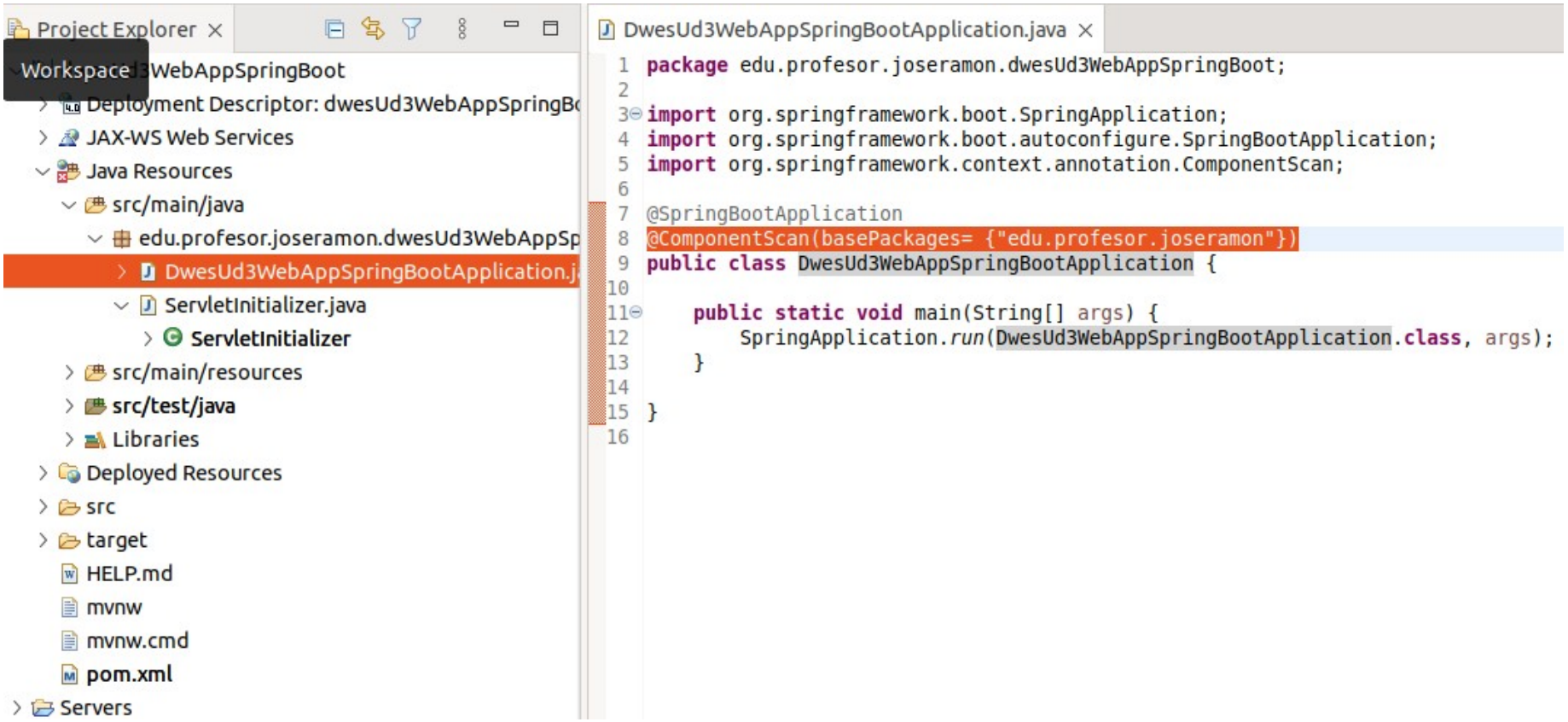
UD 3: Bases de datos y servicios REST
1.- Introducción

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



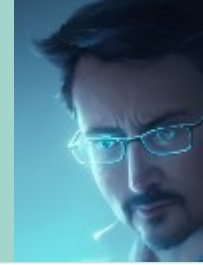
... continuación 2º Configuración del proyecto para generar HolaMundo :

Añade la notación **@ComponentScan** para decirle desde la clase principal en que paquete buscar los controladores:
Hará falta importar la librería ComponentScan...



UD 3: Bases de datos y servicios REST
1.- Introducción

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



... continuación 2º Configuración del proyecto para generar HolaMundo :

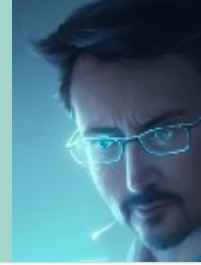
Si ya no tenemos alumno-servlet.xml, ¿Como configuramos el InternalResourceViewResolver para que sepa que los ficheros de la vista están en /WEB-INF/views/ y son ficheros con extensión JSP?

alumno-servlet.xml

```xml
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
        xmlns:context="http://www.springframework.org/schema/context"
        xmlns:mvc="http://www.springframework.org/schema/mvc"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.spri
        http://www.springframework.org/schema/mvc http://www.springframework.org/schema
        http://www.springframework.org/schema/context http://www.springframework.org/sc
        <context:component-scan base-package="org.profesor.joseramon" />
        <mvc:annotation-driven />
<bean
        class="org.springframework.web.servlet.view.InternalResourceViewResolver">
        <property name="prefix">
            <value>/WEB-INF/views/</value>
        </property>
        <property name="suffix">
            <value>.jsp</value>
        </property>
</bean>
<bean id="multipartResolver"
        class="org.springframework.web.multipart.commons.CommonsMultipartResolver">
</bean>

<!-- Internacionalización i18n -->
<bean id="messageSource"
    class="org.springframework.context.support.ReloadableResourceBundleMessageSource">
    <property name="basename" value="classpath:messages" />
    <property name="defaultEncoding" value="UTF-8" />
</bean>
<bean id="localeResolver"
        class="org.springframework.web.servlet.i18n.SessionLocaleResolver">
        <property name="defaultLocale" value="es" />
</bean>
<mvc:interceptors>
    <bean id="localeChangeInterceptor"
        class="org.springframework.web.servlet.i18n.LocaleChangeInterceptor">
        <property name="paramName" value="language" />
    </bean>
</mvc:interceptors>
<!-- Fin internacionalización i18n -->

<mvc:resources mapping="/webjars/**" location="/webjars/"/>
<mvc:resources mapping="/imagenes/**" location="/resources/imagenes/"/>
</beans>
```

UD 3: Bases de datos y servicios REST
1.- Introducción

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

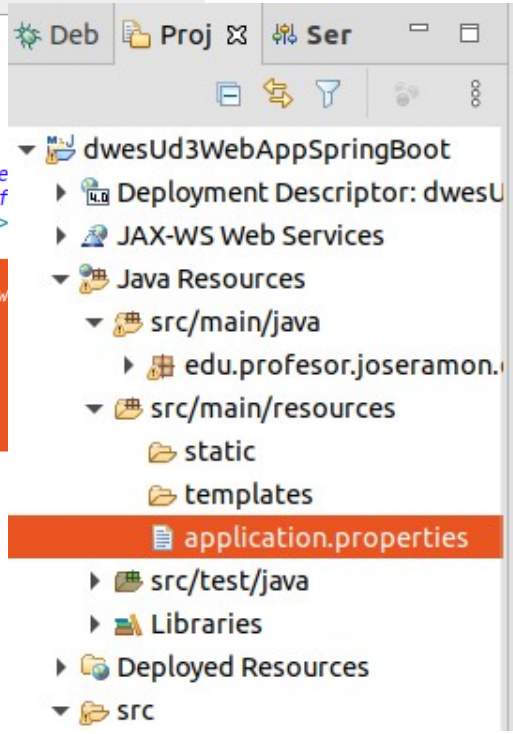


... continuación 2º Configuración del proyecto para generar HolaMundo :

Para configurar el InternalResourceViewResolver y decirle en Spring Boot que los ficheros con las vistas están en /WEB-INF/views y tienen extensión ".jsp" tan solo debemos de añadir 2 lineas con la configuración en el fichero application.properties que se ha creado automáticamente en la carpeta "src\main\resources" al generar el proyecto Spring Boot. Sin añadir estas 2 lineas, Spring Boot busca ficheros html en la carpeta "templates" en vez de ficheros "jps" en la carpeta "views" :

alumno-servlet.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
        xmlns:context="http://www.springframework.org/schema/context"
        xmlns:mvc="http://www.springframework.org/schema/mvc"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.springframework.org/schema/beans
        http://www.springframework.org/schema/mvc http://www.springframe
        http://www.springframework.org/schema/context http://www.springf
        <context:component-scan base-package="org.profesor.joseramon" />
        <mvc:annotation-driven />
<bean
        class="org.springframework.web.servlet.view.InternalResourceView
        <property name="prefix">
            <value>/WEB-INF/views/</value>
        </property>
        <property name="suffix">
            <value>.jsp</value>
        </property>
</bean>
```



application.properties

```
spring.mvc.view.prefix: /WEB-INF/views/
spring.mvc.view.suffix: .jsp
```

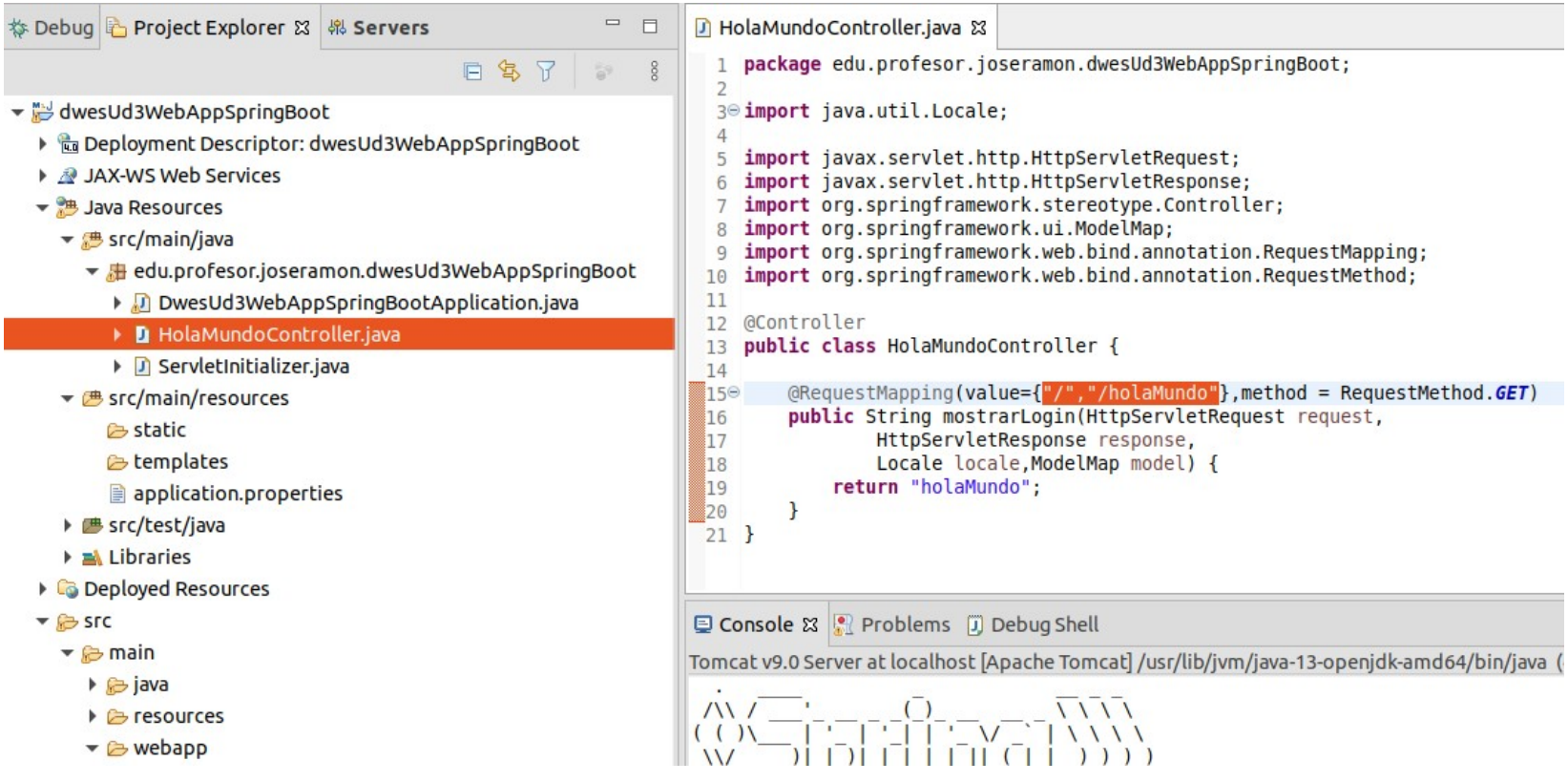
UD 3: Bases de datos y servicios REST
1.- Introducción

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

... continuación 2º Configuración del proyecto para generar HolaMundo :

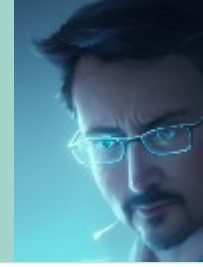
Si te has fijado, cuando hemos añadido y guardado la notación ComponentScan en el fichero java y cuando hemos modificado application.properties la aplicación se ha vuelto a cargar en la consola gracias a la dependencia “Spring Boot DevTools”.

El siguiente paso es añadir HolaMundoController para atender “/” :



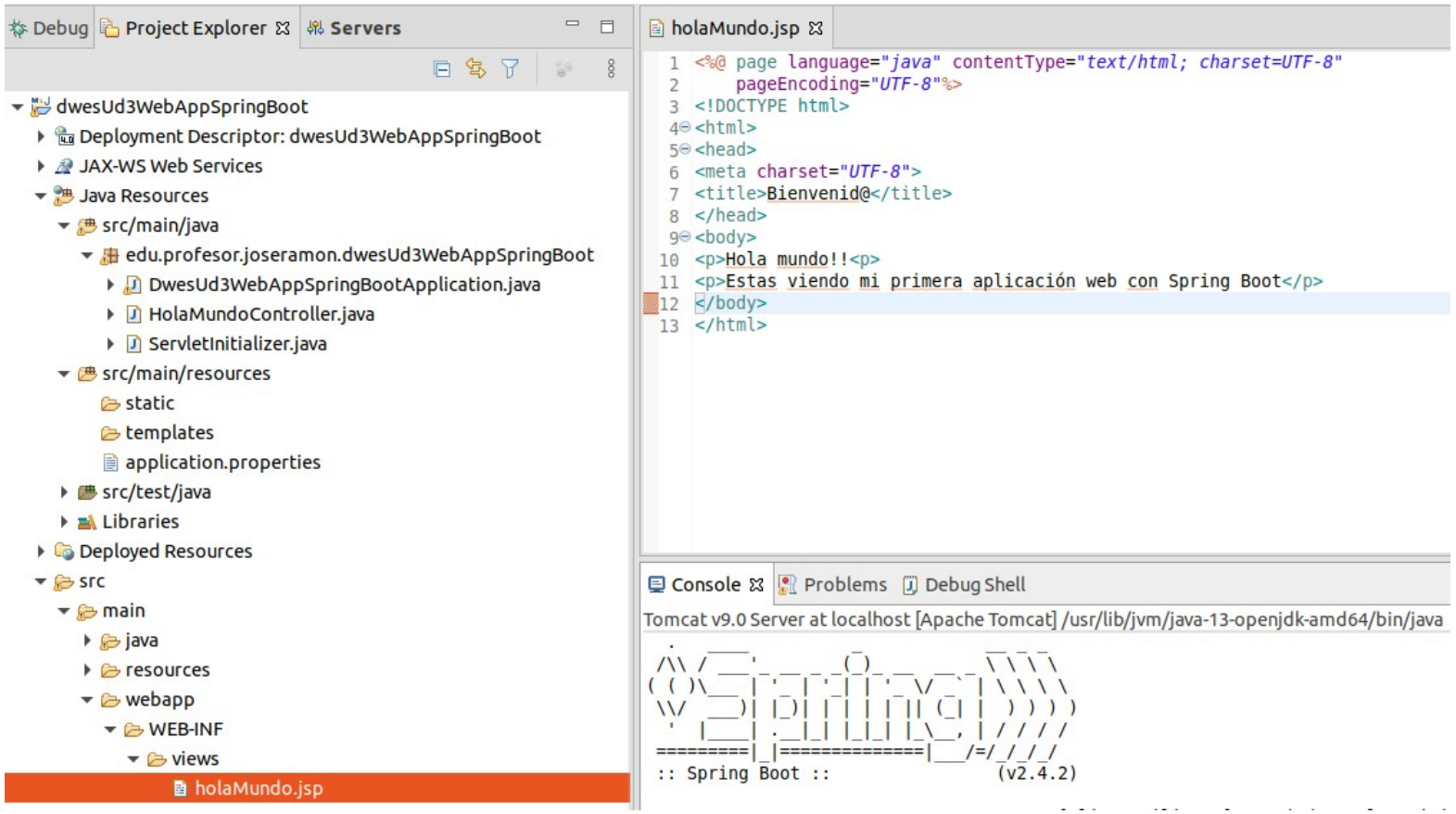
UD 3: Bases de datos y servicios REST
1.- Introducción

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



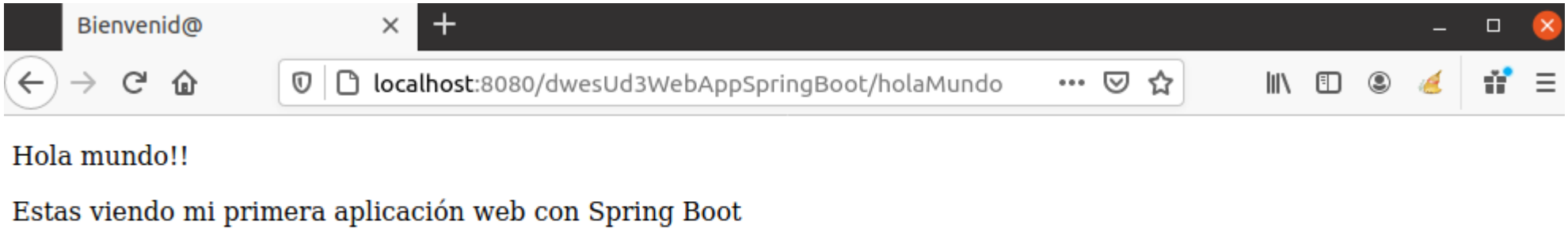
... continuación 2º Configuración del proyecto para generar HolaMundo :

Creamos la carpeta WEB-INF\views dentro de src\main\webapp. Por último debemos generar un fichero jsp nuevo dentro (holaMundo.jsp) y modificarlo para que tenga una estética similar a la siguiente:



... continuación 2º Configuración del proyecto para generar HolaMundo :

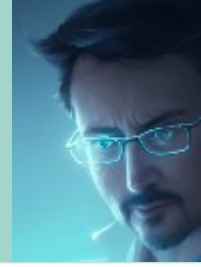
Si ponemos en marcha la web (Run As\Run on Server) podemos comprobar que todo funciona correctamente:



¿Que debemos hacer ahora para configurar el proyecto y añadirle las clases y vistas de la UD2 ?

UD 3: Bases de datos y servicios REST
1.- Introducción

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



3º Configurar el proyecto,las clases y las vistas de la UD2 :

Debemos copiar las zonas sombreadas del pom.xml de la Ud2 al nuevo proyecto para añadir las dependencias, propiedades y plugins necesarios para que funcione el proyecto antiguo con Spring Boot. No estaría de más modificar la dependencia de Lombok que generó Spring Boot para indicarle la misma versión.

dwesUd3WebAppSpringBoot/pom.xml

```
    </parent>
    <groupId>edu.profesor.joseramon</groupId>
    <artifactId>dwesUd3WebAppSpringBoot</artifactId>
    <version>0.0.1-SNAPSHOT</version>
    <packaging>war</packaging>
    <name>dwesUd3WebAppSpringBoot</name>
    <description>Aplicación web  para la Ud3 de 2º DAW DWES</description>
    <properties>
        <java.version>11</java.version>
        <org.projectlombok.version>1.18.16</org.projectlombok.version>
        <org.mapstruct.version>1.4.1.Final</org.mapstruct.version>
    </properties>
    <dependencies>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-data-jpa</artifactId>
```

dwesUd3WebAppSpringBoot/pom.xml

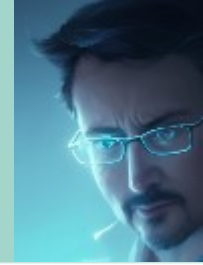
```
        <plugins>
            <plugin>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-maven-plugin</artifactId>
                <configuration>
                    <excludes>
                        <exclude>
                            <groupId>org.projectlombok</groupId>
                            <artifactId>lombok</artifactId>
                        </exclude>
                    </excludes>
                </configuration>
            </plugin>
            <plugin>
                <groupId>org.apache.maven.plugins</groupId>
                <artifactId>maven-compiler-plugin</artifactId>
                <version>3.8.1</version>
                <configuration>
                    <verbose>true</verbose>
                    <source>1.8</source>
                    <target>1.8</target>
                    <encoding>UTF-8</encoding>
                    <showWarnings>true</showWarnings>
                    <annotationProcessorPaths>
                        <path>
                            <groupId>org.projectlombok</groupId>
                            <artifactId>lombok</artifactId>
                            <version>${org.projectlombok.version}</version>
                        </path>
                        <!-- Necesario para que Lombok funcione con MapStruct -->
                        <path>
                            <groupId>org.projectlombok</groupId>
                            <artifactId>lombok-mapstruct-binding</artifactId>
                            <version>0.2.0</version>
                        </path>
                        <path>
                            <groupId>org.mapstruct</groupId>
                            <artifactId>mapstruct-processor</artifactId>
                            <version>${org.mapstruct.version}</version>
                        </path>
                        <!-- other annotation processors -->
                    </annotationProcessorPaths>
                </configuration>
            </plugin>
        </plugins>
    </build>
</project>
```

dwesUd3WebAppSpringBoot/pom.xml

```
            <scope>provided</scope>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-test</artifactId>
            <scope>test</scope>
        </dependency>
        <!--  JSTL  -->
        <dependency>
            <groupId>jstl</groupId>
            <artifactId>jstl</artifactId>
            <version>1.2</version>
        </dependency>
        <!--  Css Bootstrap  -->
        <dependency>
            <groupId>org.webjars</groupId>
            <artifactId>bootstrap</artifactId>
            <version>4.5.2</version>
        </dependency>
        <!--  iconos fuentes awesome  -->
        <dependency>
            <groupId>org.webjars</groupId>
            <artifactId>font-awesome</artifactId>
            <version>5.15.1</version>
        </dependency>
        <!--  Apache Commons FileUpload  -->
        <dependency>
            <groupId>commons-fileupload</groupId>
            <artifactId>commons-fileupload</artifactId>
            <version>1.4</version>
        </dependency>
        <!--  Apache Commons IO  -->
        <dependency>
            <groupId>commons-io</groupId>
            <artifactId>commons-io</artifactId>
            <version>2.8.0</version>
        </dependency>
        <!--  DTOs  -->
        <dependency>
            <groupId>org.mapstruct</groupId>
            <artifactId>mapstruct</artifactId>
            <version>${org.mapstruct.version}</version>
            <scope>compile</scope>
        </dependency>
    </dependencies>
    <build>
        <plugins>
            <plugin>
```

UD 3: Bases de datos y servicios REST
1.- Introducción

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



... continuación 3º Configurar el proyecto,las clases y las vistas de la UD2 :

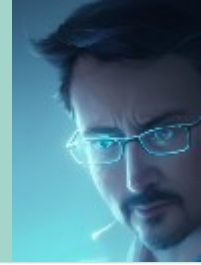
¿Que debemos hacer ahora para configurar los beans y las carpetas con recursos que habían en alumno-servlet.xml?

alumno-servlet.xml

```xml
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
        xmlns:context="http://www.springframework.org/schema/context"
        xmlns:mvc="http://www.springframework.org/schema/mvc"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.spri
        http://www.springframework.org/schema/mvc http://www.springframework.org/schema
        http://www.springframework.org/schema/context http://www.springframework.org/sc
        <context:component-scan base-package="org.profesor.joseramon" />
        <mvc:annotation-driven />
<bean
        class="org.springframework.web.servlet.view.InternalResourceViewResolver">
        <property name="prefix">
            <value>/WEB-INF/views/</value>
        </property>
        <property name="suffix">
            <value>.jsp</value>
        </property>
</bean>
<bean id="multipartResolver"
        class="org.springframework.web.multipart.commons.CommonsMultipartResolver">
</bean>

<!-- Internacionalización i18n -->
<bean id="messageSource"
    class="org.springframework.context.support.ReloadableResourceBundleMessageSource">
    <property name="basename" value="classpath:messages" />
    <property name="defaultEncoding" value="UTF-8" />
</bean>
<bean id="localeResolver"
        class="org.springframework.web.servlet.i18n.SessionLocaleResolver">
        <property name="defaultLocale" value="es" />
</bean>
<mvc:interceptors>
    <bean id="localeChangeInterceptor"
        class="org.springframework.web.servlet.i18n.LocaleChangeInterceptor">
        <property name="paramName" value="language" />
    </bean>
</mvc:interceptors>
<!-- Fin internacionalización i18n -->

<mvc:resources mapping="/webjars/**" location="/webjars/"/>
<mvc:resources mapping="/imagenes/**" location="/resources/imagenes/"/>
</beans>
```

UD 3: Bases de datos y servicios REST
1.- Introducción

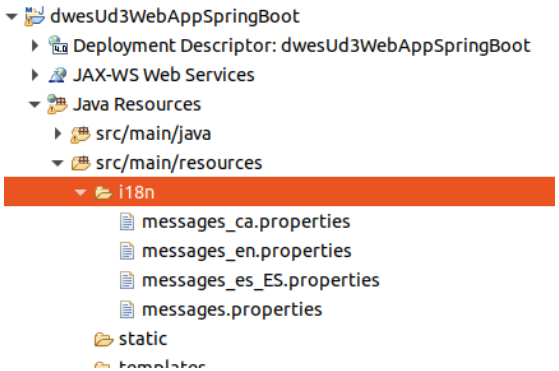
Desarrollo Web en Entorno Servidor - joseramon.profesor@gmail.com



... continuación 3º Configurar el proyecto,las clases y las vistas de la UD2 :

Realmente podríamos dejar el mismo fichero web.xml y alumno-servlet.xml retocando algunas cosas, pero queremos aprovechar las notaciones de Spring Boot para crear una clase nueva (que se proporciona en el Drive) con la cual podemos configurar de forma centralizada nuestra WebApp y realizar la misma configuración del xml desde una clase.

Es importante destacar que deberemos de copiar los ficheros de traducción a una carpeta I18 tal cual se vé decimos en la configuración:



AppConfig.java

```
1  package edu.profesor.joseramon.dwesUd3WebAppSpringBoot;
2
3  import java.io.IOException;
18
19 @Configuration
20 @ComponentScan(basePackages={"edu.profesor.joseramon"})
21 public class AppConfig implements WebMvcConfigurer {
22
23     @Bean
24     public UrlBasedViewResolver viewResolver() {
25         UrlBasedViewResolver resolver
26           = new UrlBasedViewResolver();
27         resolver.setPrefix("/WEB-INF/views/");
28         resolver.setSuffix(".jsp");
29         resolver.setViewClass(JstlView.class);
30         return resolver;
31     }
32
33     @Bean
34     public CommonsMultipartResolver multipartResolver()
35       throws IOException {
36         CommonsMultipartResolver resolver
37           = new CommonsMultipartResolver();
38         resolver.setMaxUploadSize(10000000);
39         return resolver;
40     }
41
42     @Bean
43     public ReloadableResourceBundleMessageSource messageSource() {
44         ReloadableResourceBundleMessageSource  resource = new Reloa
45         resource.setBasename("classpath:i18n/messages");
46         resource.setDefaultEncoding("UTF-8");
47 //        resource.setCacheSeconds(10);
48         return resource;
49     }
50
51     @Bean
52     public LocaleResolver localeResolver() {
53         CookieLocaleResolver cl = new CookieLocaleResolver();
54         cl.setCookieName("language");
55         return cl;
56     }
```

UD 3: Bases de datos y servicios REST
1.- Introducción

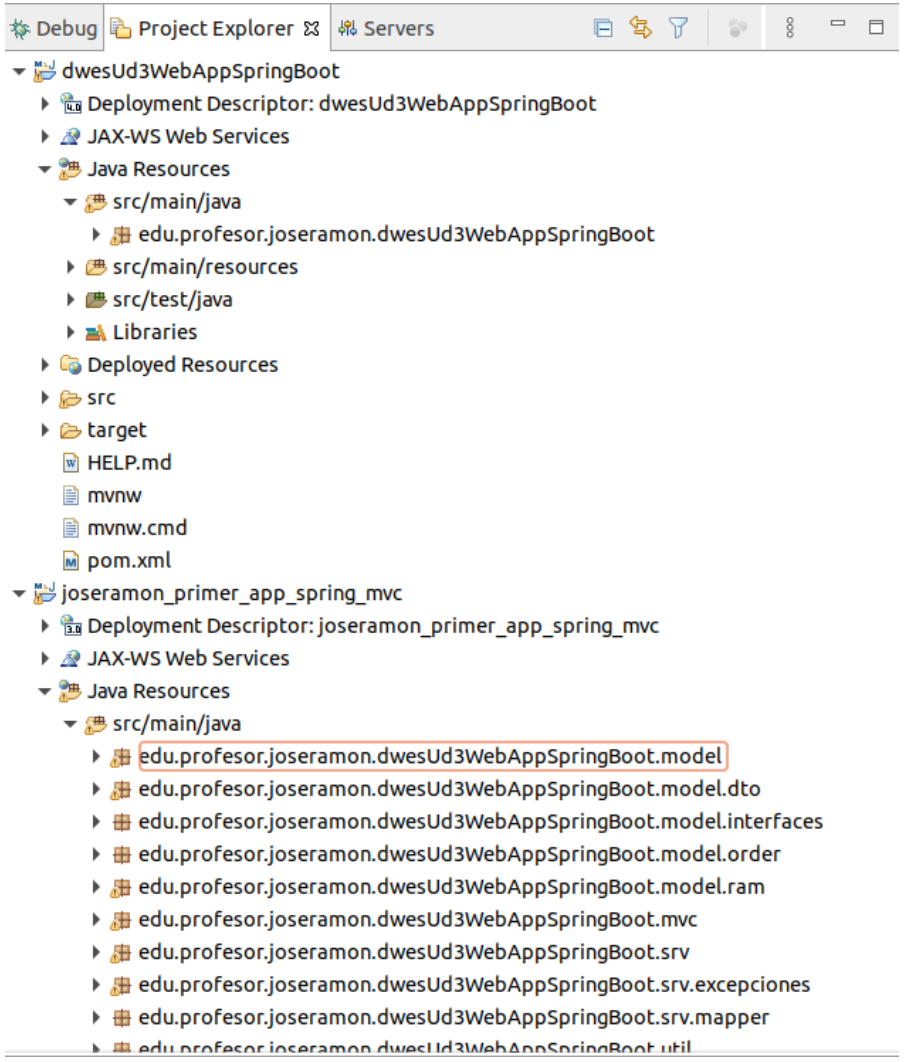
Desarrollo Web en Entorno Servidor - joseramon.profesor@gmail.com

... continuación 3º Configurar el proyecto,las clases y las vistas de la UD2 :

Para pasarle al proyecto Spring Boot todos los paquetes del proyecto de la Ud 2 podemos realizar los siguientes pasos:

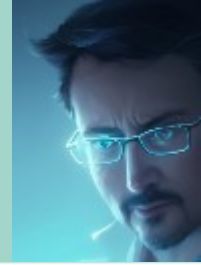
1º Hacer una copia de seguridad del proyecto de la Ud 2 porqué por rapidez vamos a modificarlo y se quedará inconsistente.

2º Para evitar tener que cambiarle el package a todas las clases lo que haremos será refactorizar todos los paquetes del proyecto de la Ud 2 (botón derecho sobre cada paquete y refactor\rename) para ponerles el nombre adecuado y poder reutilizarlos en el nuevo proyecto.



UD 3: Bases de datos y servicios REST
1.- Introducción

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



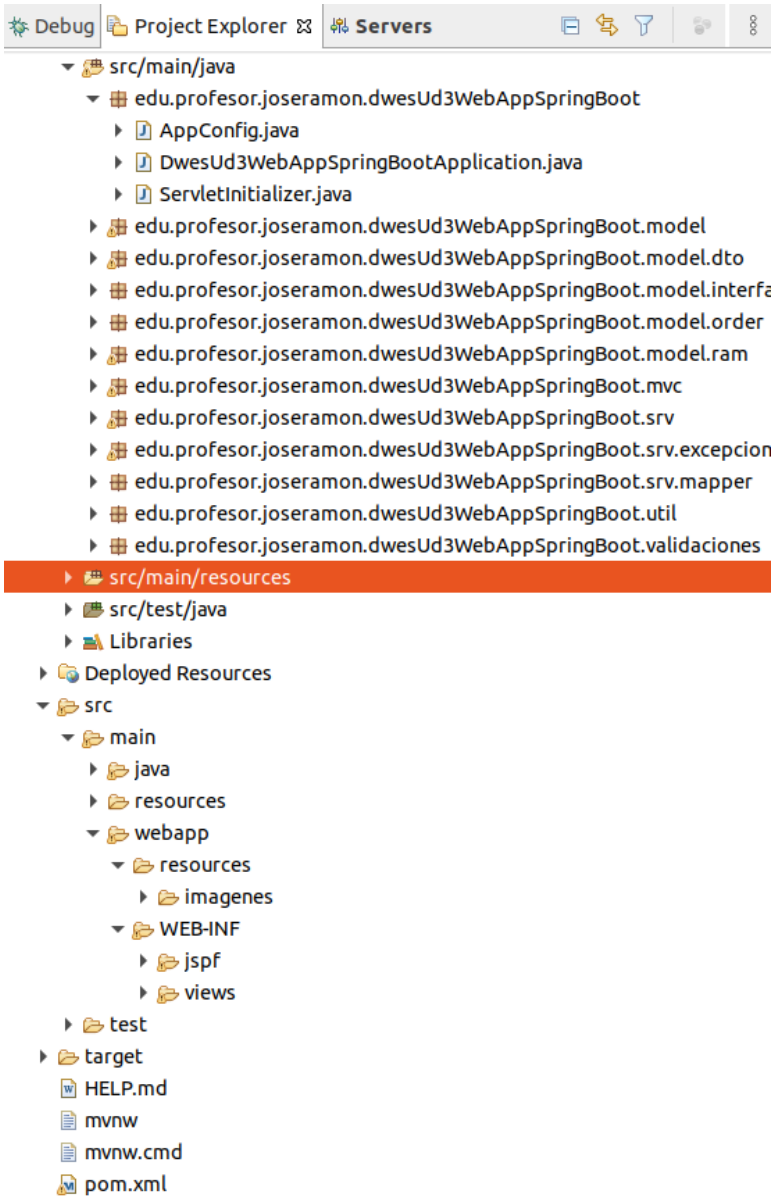
... continuación 3º Configurar el proyecto,las clases y las vistas de la UD2 :

3º Salimos de Eclipse, comprimimos la carpeta edu del proyecto de la Ud2 y lo copiamos al proyecto Spring Boot.

4º Abrimos Eclipse y podemos comprobar que los paquetes se ven en el nuevo proyecto. Acordarse de borrar HolaMundoController o entrará en conflicto con LoginController.

5º Para copiar los ficheros jsp es más sencillo, simplemente copiar la carpeta WEB-INF teniendo la precaución de borrar web.xml y alumno-servlet.xml después. Copiar también la carpeta webapp\resources\imagenes.

Ahora se entiende porque se suelen hacer proyectos que son módulos importables a otros proyectos, porqué así no se tienen que hacer tantos cambios, solo importar el módulo.



UD 3: Bases de datos y servicios REST
1.- Introducción

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

... continuación 3º Configurar el proyecto,las clases y las vistas de la UD2 :

Tenemos que realizar una pequeña modificación en el servicio de internacionalización. Realizar el cambio:

I18nService.java

```
 9 import java.util.Set;
10
11 import javax.servlet.http.HttpServletRequest;
12 import javax.servlet.http.HttpServletResponse;
13
14 import org.springframework.beans.factory.annotation.Autowired;
15 import org.springframework.context.NoSuchMessageException;
16 import org.springframework.context.i18n.LocaleContextHolder;
17 import org.springframework.context.support.ReloadableResourceBundleMessageSourc
18 import org.springframework.stereotype.Service;
19 import org.springframework.web.servlet.LocaleResolver;
20 @Service
21 public class I18nService {
22     @Autowired
23     private ReloadableResourceBundleMessageSource i18n_mensaje;
24     @Autowired
25     //Con Spring MVC: private SessionLocaleResolver idiomaPeticiones; //bean de
26     /Con Spring Boot cambia la clase de idiomaPeticiones
27     private LocaleResolver idiomaPeticiones;
28     //Consulta el idioma configurado en la peticion y devuelve su traducción
29     private String getI18nMessage(String msg) {
30         if (msg.contains("#")) {
```

UD 3: Bases de datos y servicios REST
1.- Introducción

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

... continuación 3º Configurar el proyecto,las clases y las vistas de la UD2 :

El motivo de este cambio es que con una aplicación Spring MVC si no configuramos LocaleResolver, utiliza por defecto AcceptHeaderLocaleResolver, que no permite cambiar locale.

Para solucionarlo en la UD2 creamos un bean SessionLocaleResolver.

Sin embargo, con Spring Boot da error si creamos un bean de tipo SessionLocaleResolver y luego utilizando “language” lo cambiamos, por lo que gastamos cookies que utilizamos en la sesión (CookieLocaleResolver).

AppConfig.java

```
49      }
50
51      /**
52       * Bean con Spring MVC:
53      @Bean
54      public SessionLocaleResolver sessionLocaleResolver() {
55          SessionLocaleResolver localeResolver = new SessionLocaleResolver();
56          localeResolver.setDefaultLocale(Locale.getDefault());
57          //localeResolver.setDefaultLocale(new Locale("es"));
58          localeResolver.setDefaultTimeZone(TimeZone.getDefault());
59          return localeResolver;
60      }
61      Spring Boot trabaja diferente a Spring MVC:
62      Con una aplicación Spring MVC si no configuramos LocaleResolver, utiliza
63      por defecto AcceptHeaderLocaleResolver, que no permite cambiar locale.
64      Para solucionarlo en la UD2 creamos un bean SessionLocaleResolver.
65      Sin embargo, con Spring Boot da error si creamos un bean de tipo SessionLocaleResolver,
66      por lo que gastamos cookies que utilizamos en la sesión (CookieLocaleResolver).
67      Mas info en: https://www.programmersought.com/article/17992294758/
68      **/
69      @Bean
70      public LocaleResolver localeResolver() {
71          CookieLocaleResolver cl = new CookieLocaleResolver();
72          cl.setCookieName("language");
73          return cl;
74      }
75
76      @Bean
77      public LocaleChangeInterceptor localeChangeInterceptor() {
78          LocaleChangeInterceptor localeChangeInterceptor = new LocaleChangeInterceptor();
```

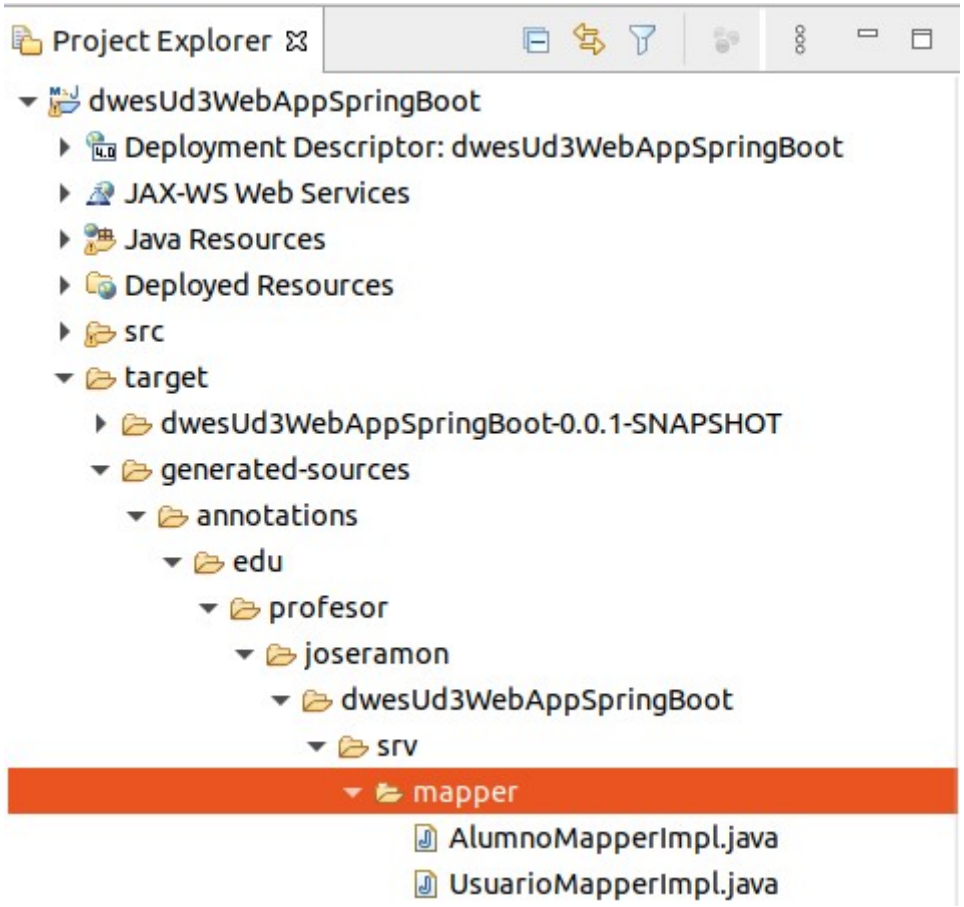
Mas información en: https://www.programmersought.com/article/17992294758/

UD 3: Bases de datos y servicios REST
1.- Introducción

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

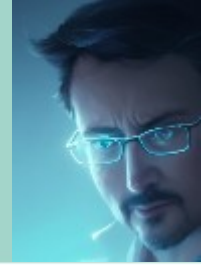
... continuación 3º Configurar el proyecto,las clases y las vistas de la UD2 :

Como hemos copiado el proyecto de la UD2 debemos asegurarnos de hacer un "Run as\Maven clean", un "Maven\Update project" y *para asegurarnos que se generan las clases de mapper "Run as\Maven install"*:



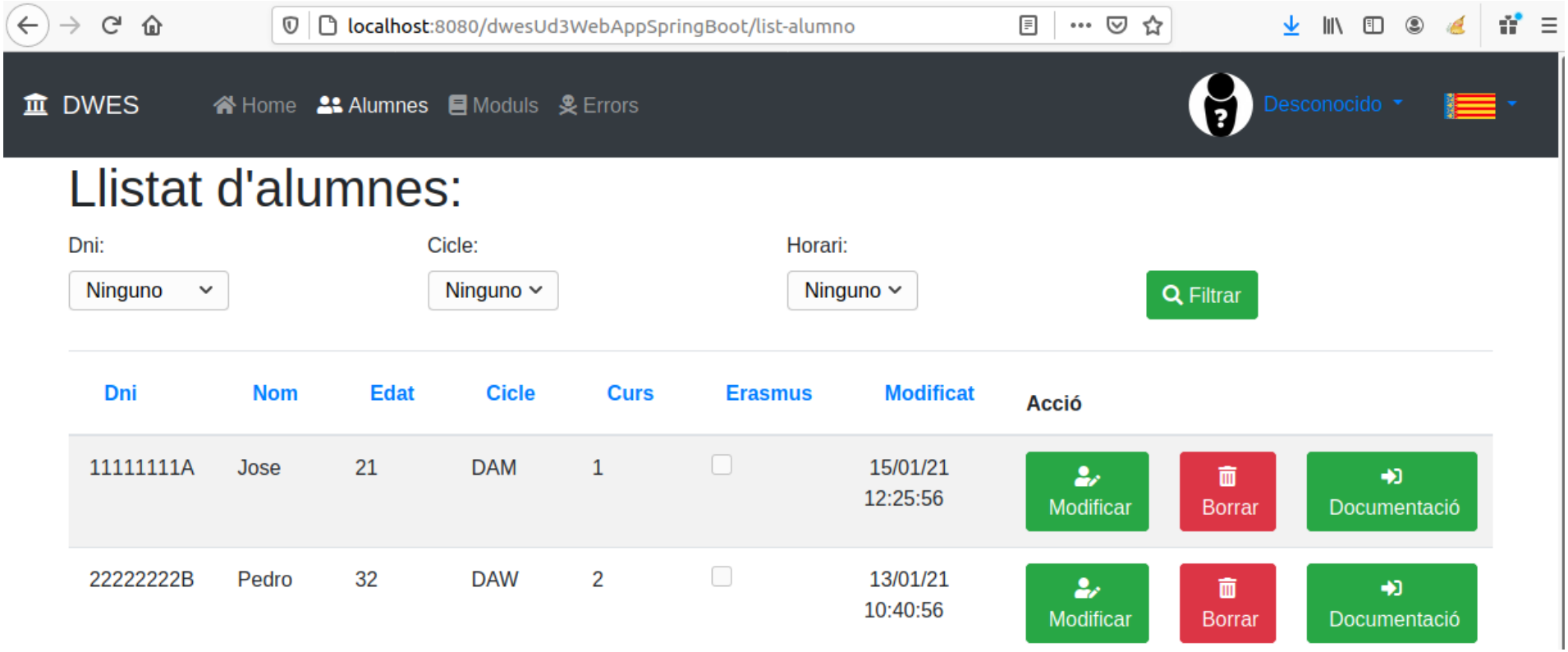
UD 3: Bases de datos y servicios REST
1.- Introducción

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



... continuación 3º Configurar el proyecto,las clases y las vistas de la UD2 :

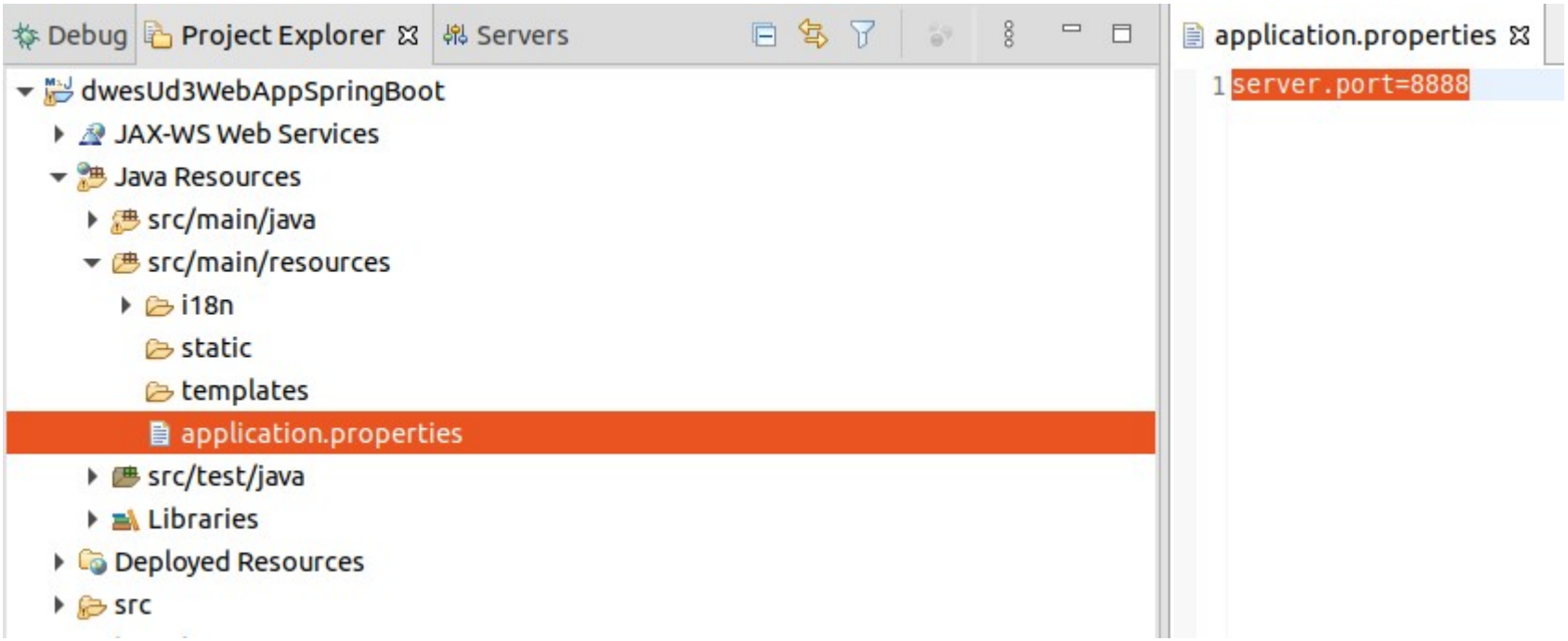
Solo nos queda comprobar que nuestra aplicación funciona correctamente:



Arrancar nuestra webApp Spring Boot desde linea de comandos:

Podemos instalar nuestra webApp en formato WAR en un contenedor de aplicaciones como Tomcat, pero si queremos arrancar nuestra webApp sin estar dentro de Eclipse ni cargar el War en Tomcat primero que nada cambiaremos el puerto de nuestra aplicación en el fichero de propiedades.

De esta manera conseguiremos que no se solape con cualquier Tomcat que tengamos instalado. Deberemos de hacer un “Run As\Maven clean”, “Maven\Update project”, un “Run As\Maven install” y cerrar Eclipse:



UD 3: Bases de datos y servicios REST
1.- Introducción

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

... continuación Arrancar nuestra webApp Spring Boot desde linea de comandos:

Por otra parte, las aplicaciones web que contienen páginas jsp necesitan tener 2 dependencias más cuando se ejecutan sin ningún servidor de Tomcat. Añade las dependencias:

dwesUd3WebAppSpringBoot/pom.xml

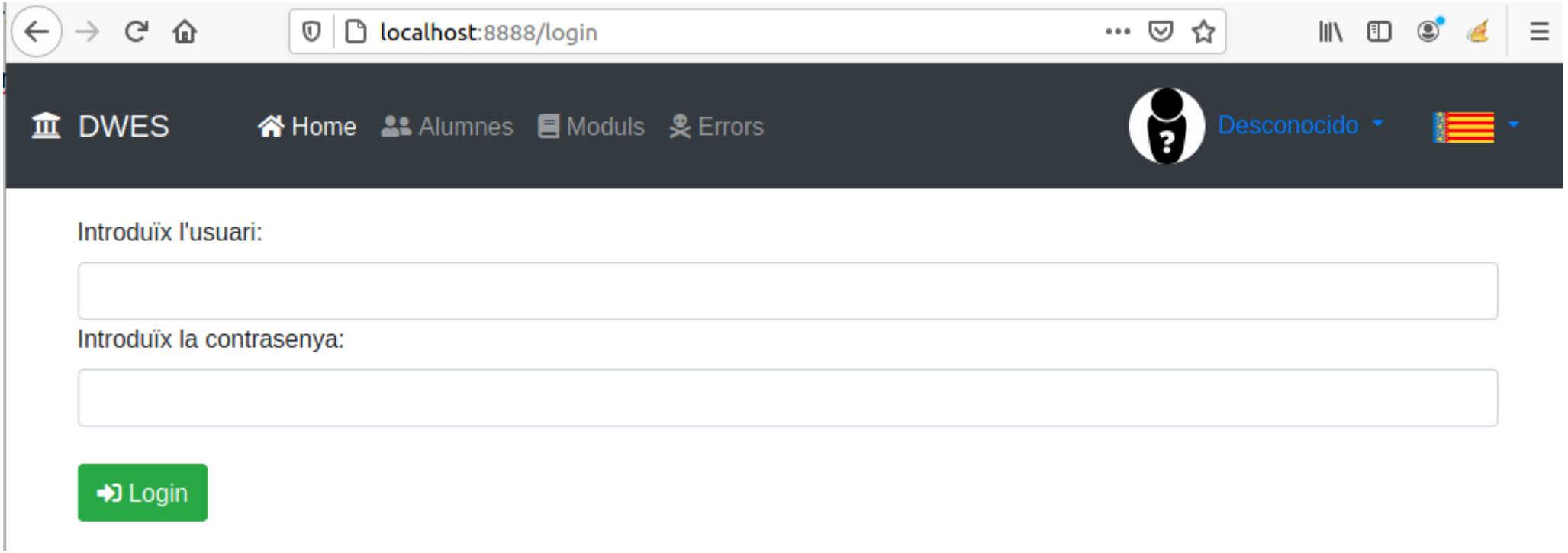
```
98              <groupId>commons-io</groupId>
99              <artifactId>commons-io</artifactId>
100             <version>2.8.0</version>
101         </dependency>
102         <!--  DTOs  -->
103         <dependency>
104             <groupId>org.mapstruct</groupId>
105             <artifactId>mapstruct</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>
```

UD 3: Bases de datos y servicios REST
1.- Introducción

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

... continuación Arrancar nuestra webApp Spring Boot desde linea de comandos:

Ahora, para arrancar nuestra webApp tenemos 2 opciones que hacen lo mismo, pero cuidado, puede que ya **no tengamos en la url el nombre de la aplicación dependiendo de como arranquemos la webapp:**



¿Cuales son estas 2 opciones?

UD 3: Bases de datos y servicios REST
1.- Introducción

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

... continuación Arrancar nuestra webApp Spring Boot desde linea de comandos:

1º opción: Utilizar maven y luego acceder a localhost sin el nombre de la webapp:

```
joseramon@Notebook-PC:~/ProyectosSpringBoot/dwesUd3WebAppSpringBoot$ mvn clean spring-boot:run
[INFO] Scanning for projects...
[INFO]
[INFO] -----------< edu.profesor.joseramon:dwesUd3WebAppSpringBoot >-----------
[INFO] Building dwesUd3WebAppSpringBoot 0.0.1-SNAPSHOT
[INFO] --------------------------------[ war ]---------------------------------
[INFO]
[INFO] --- maven-clean-plugin:3.1.0:clean (default-clean) @ dwesUd3WebAppSpringBoot ---
[INFO] Deleting /home/joseramon/ProyectosSpringBoot/dwesUd3WebAppSpringBoot/target
[INFO]
[INFO] >>> spring-boot-maven-plugin:2.4.2:run (default-cli) > test-compile @ dwesUd3WebAppSpringBoot >>>
[INFO]
[INFO] --- maven-resources-plugin:3.2.0:resources (default-resources) @ dwesUd3WebAppSpringBoot ---
[INFO] Using 'UTF-8' encoding to copy filtered resources.
[INFO] Using 'UTF-8' encoding to copy filtered properties files.
[INFO] Copying 1 resource
[INFO] Copying 4 resources
[INFO]
[INFO] --- maven-compiler-plugin:3.8.1:compile (default-compile) @ dwesUd3WebAppSpringBoot ---
[INFO] Changes detected - recompiling the module!
[INFO] Compiling 60 source files to /home/joseramon/ProyectosSpringBoot/dwesUd3WebAppSpringBoot/target/classes
```

UD 3: Bases de datos y servicios REST
1.- Introducción

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

... continuación Arrancar nuestra webApp Spring Boot desde linea de comandos:

2º opción: Generar un fichero WAR y ejecutarlo:

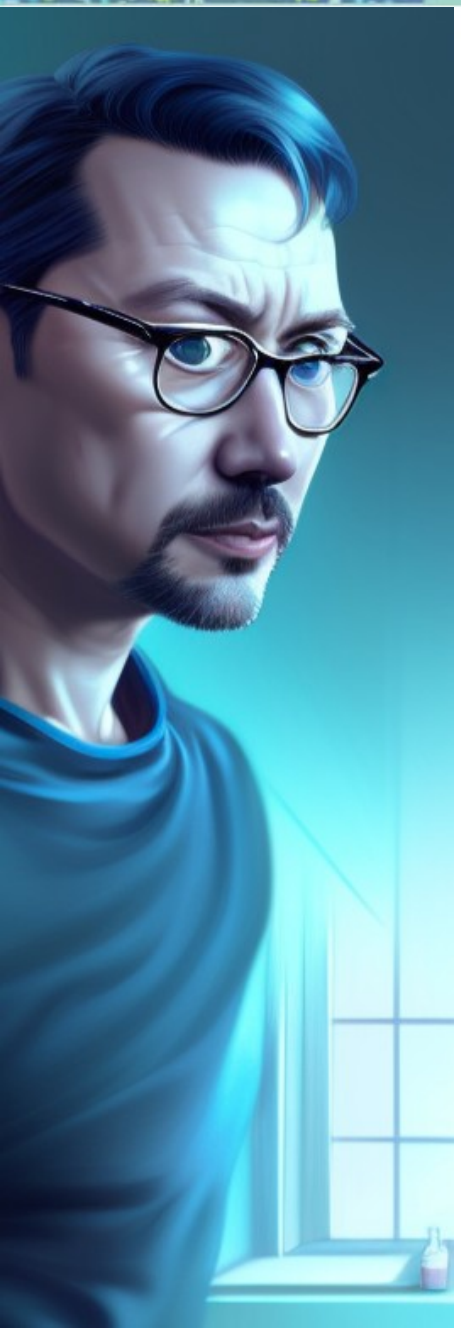
```
joseramon@Notebook-PC:~/ProyectosSpringBoot/dwesUd3WebAppSpringBoot$ mvn clean install
[INFO] Scanning for projects...
[INFO]
[INFO] -----------< edu.profesor.joseramon:dwesUd3WebAppSpringBoot >-----------
[INFO] Building dwesUd3WebAppSpringBoot 0.0.1-SNAPSHOT
[INFO] --------------------------------[ war ]---------------------------------
```

```
joseramon@Notebook-PC:~/ProyectosSpringBoot/dwesUd3WebAppSpringBoot$ java -jar target/dwesUd3WebAppSpringBoot-0.0.1-SNAPSHOT.war

  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/
 :: Spring Boot ::                (v2.4.2)
```

OJO: La gran ventaja de Spring Boot es que podemos copiar dicho war generado al realizar el “install” donde queramos y ejecutarlo sin necesidad de tener el proyecto entero ni un Tomcat instalado.

UD 3: Bases de datos y servicios REST
1.- Introducción

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_practica1_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.