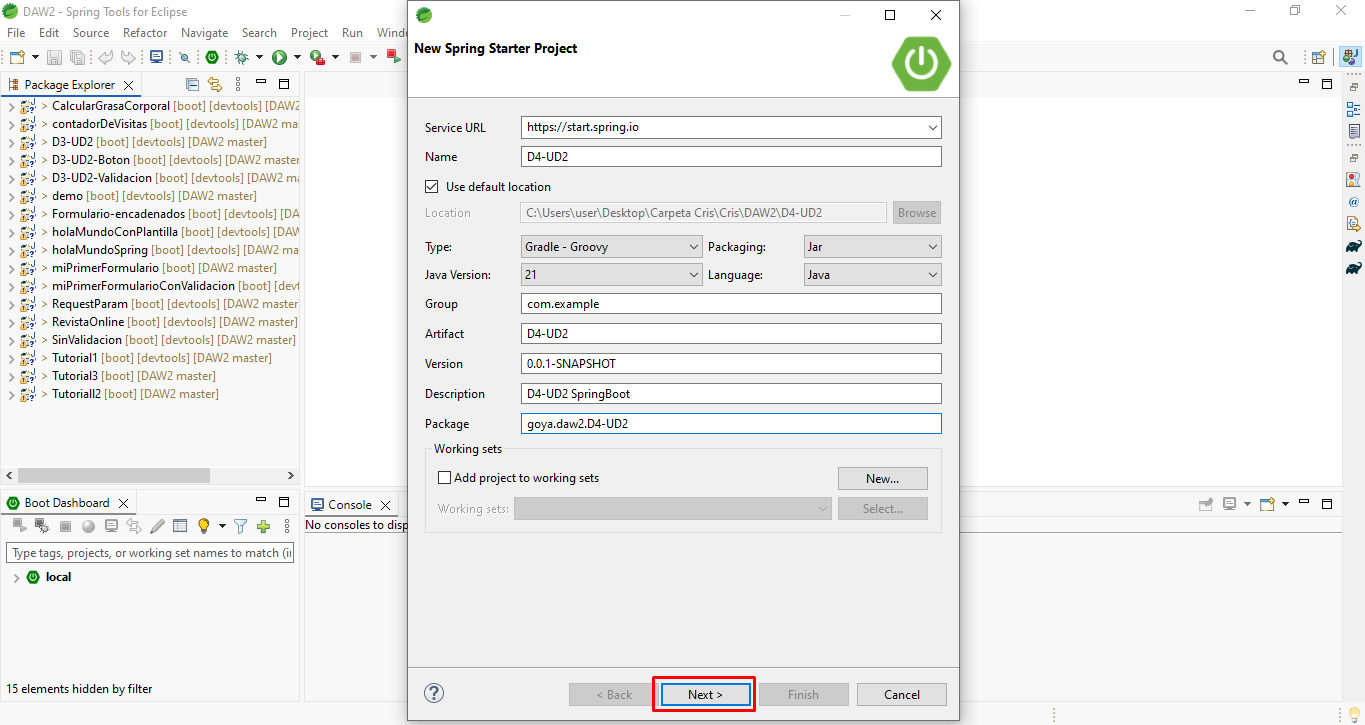
**D4-UD2. Ejercicio fin UD2**

Para rematar la UD2 os voy a pedir estos dos ejercicios:

1. Pruebas de codificación: se trata de ver cómo se comporta un formulario cualquiera al meter en un campo de texto caracteres como acentos, eñes, backslash (\) , slah(/) , comillas simples, dobles e inversas, etiquetas html, etc. En los siguientes contextos:
   1. Por POST examinando las cabeceras Content-type, Content-encoding o similares en petición y respuesta.
   2. Poniendo el formulario con GET y viendo en la URL (query-string) cómo se transforman (codifican) los valores.
   3. En vez de formulario poniendo un enlace con query-string (?var=valor&var2=valor2... , ej, ir a otra página pasando un nombre, como en el primer tutorial del saludo) y ver cómo hay que codificarlos para que se reciben bien en el lado del Mapping textos con los caracteres del punto 1. Pista: mira el método URLEncoder.encode

**Creamos un proyecto llamado D4-UD2**

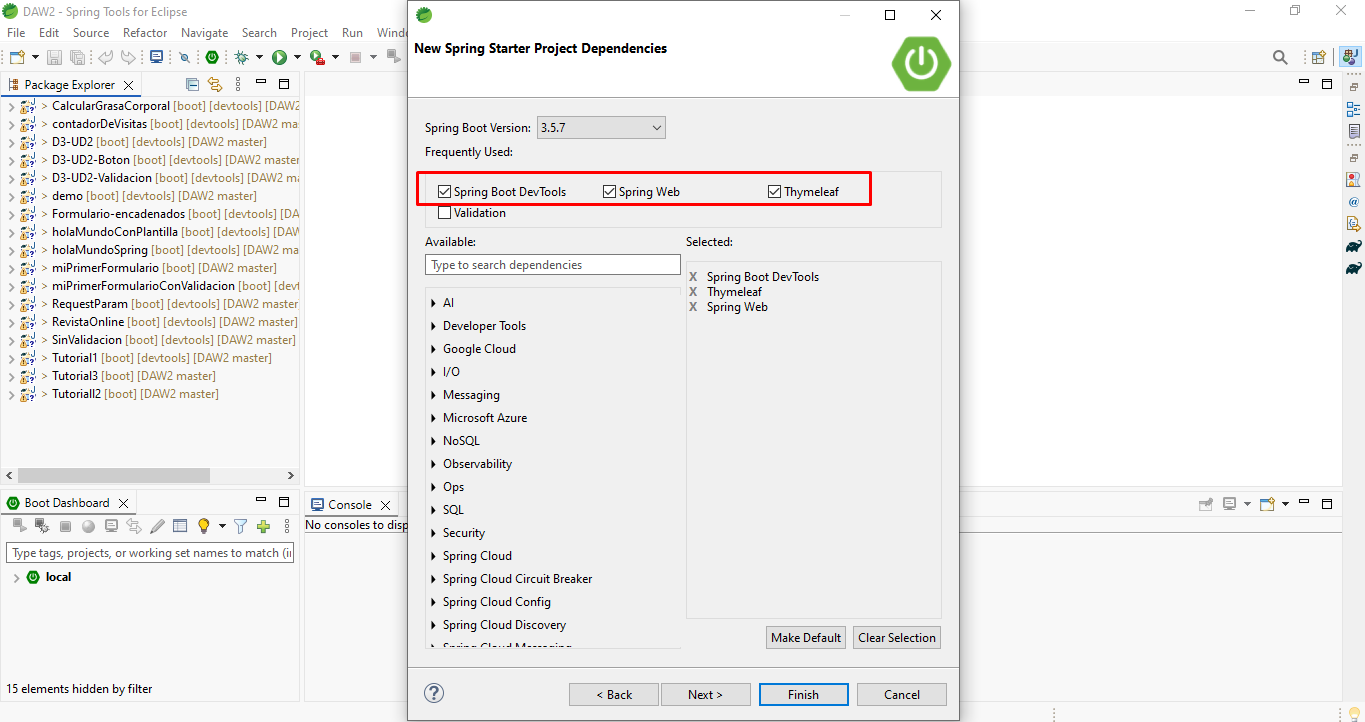


**Ponemos las 3 dependencias que necesitamos:**

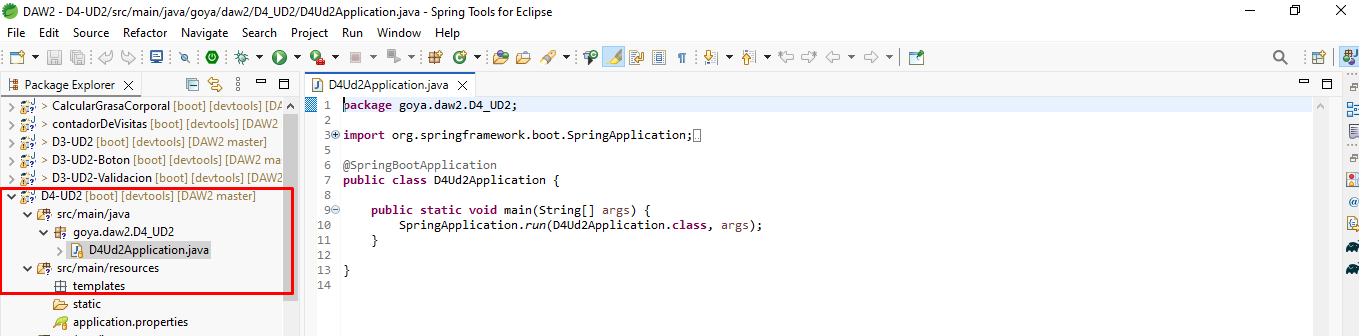
**Spring web: para controladores y manejo de formularios GET/POST**

**Thymeleaf: para las plantillas HTML**

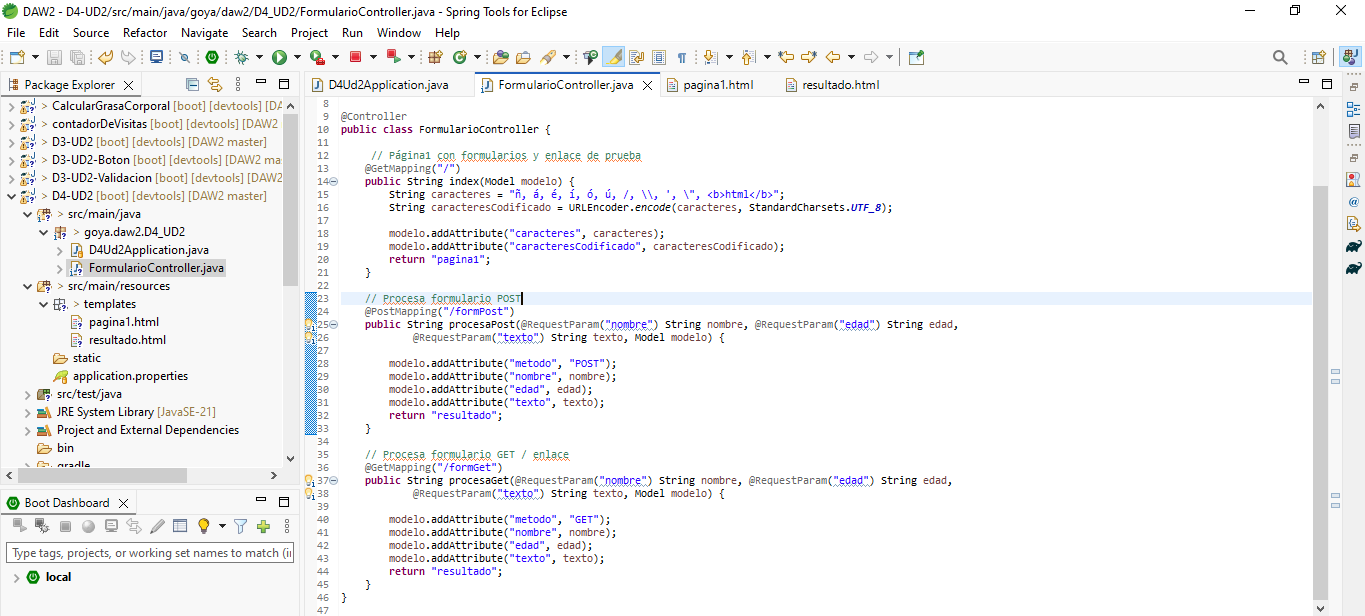
**DevTools: para cambios y reinicios del proyecto**



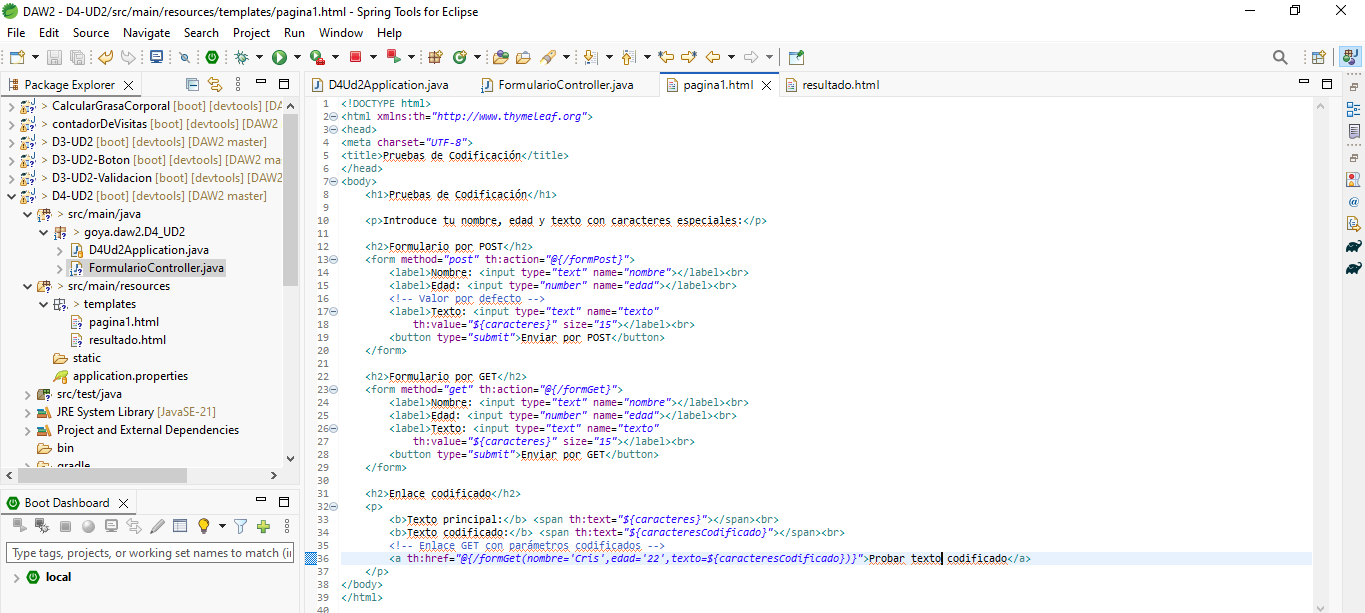
**Vemos que nos ha creado el proyecto y la clase por defecto**



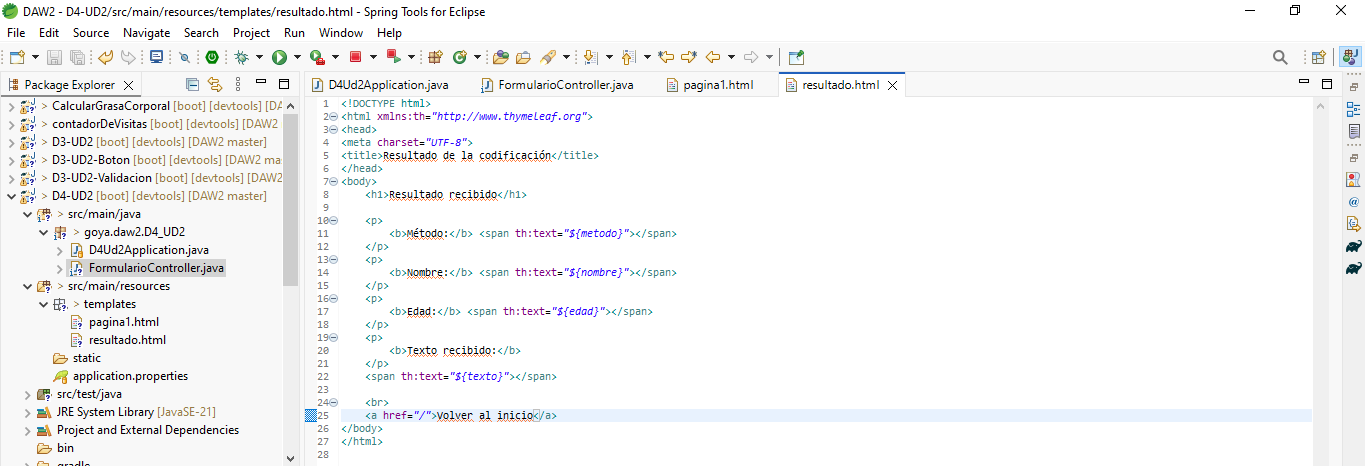
**Creamos la clase FormularioController, se encarga de recibir los datos que el usuario introduce en los formularios y mostrarlos en pantalla. Su objetivo es comprobar cómo se codifican los textos con símbolos especiales cuando se envían al servidor mediante los métodos GET o POST.**



**Creamos el HTML llamado pagina1.html, que muestra los formularios y un enlace para realizar las pruebas de codificación. Permite al usuario introducir su nombre, edad y un texto con caracteres especiales y enviarlos al servidor mediante los métodos POST o GET. Además, incluye un enlace con los datos ya codificados, para observar cómo se representan los caracteres especiales en la URL.**

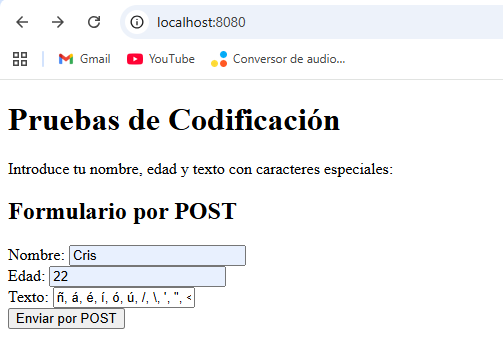


**Creamos el HTML llamado resultado.html, que muestra los datos enviados desde el formulario. En esta página se visualizan el método utilizado (GET o POST), el nombre, la edad y el texto introducido por el usuario, permitiendo comprobar cómo se han recibido y decodificado los caracteres especiales en el servidor.**



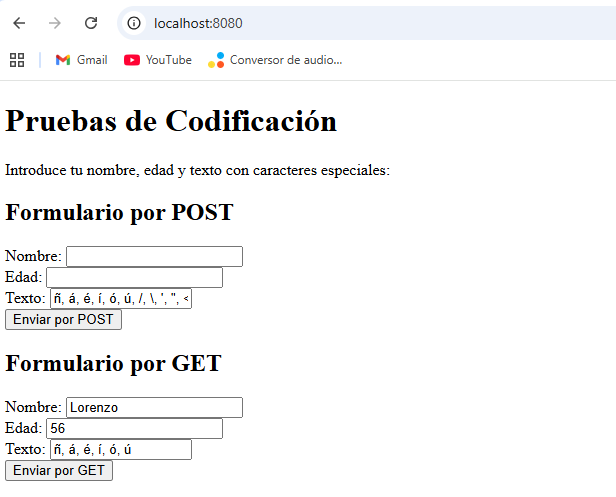
**En el navegador:** [**http://localhost:8080/**](http://localhost:8080/)

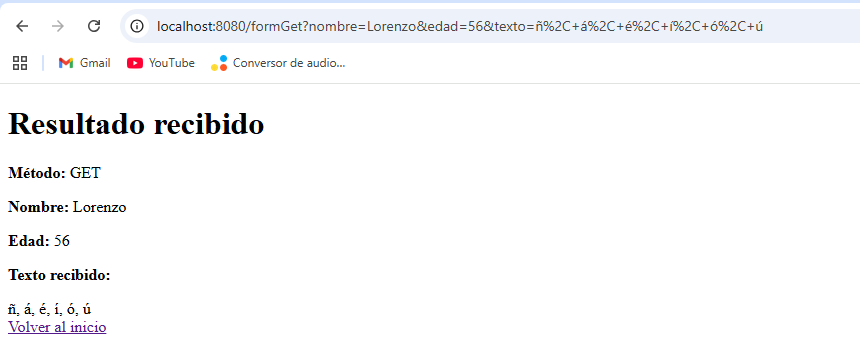
**Al usar el formulario por POST, escribimos nuestro nombre, la edad, un texto y luego hacemos clic en Enviar.**



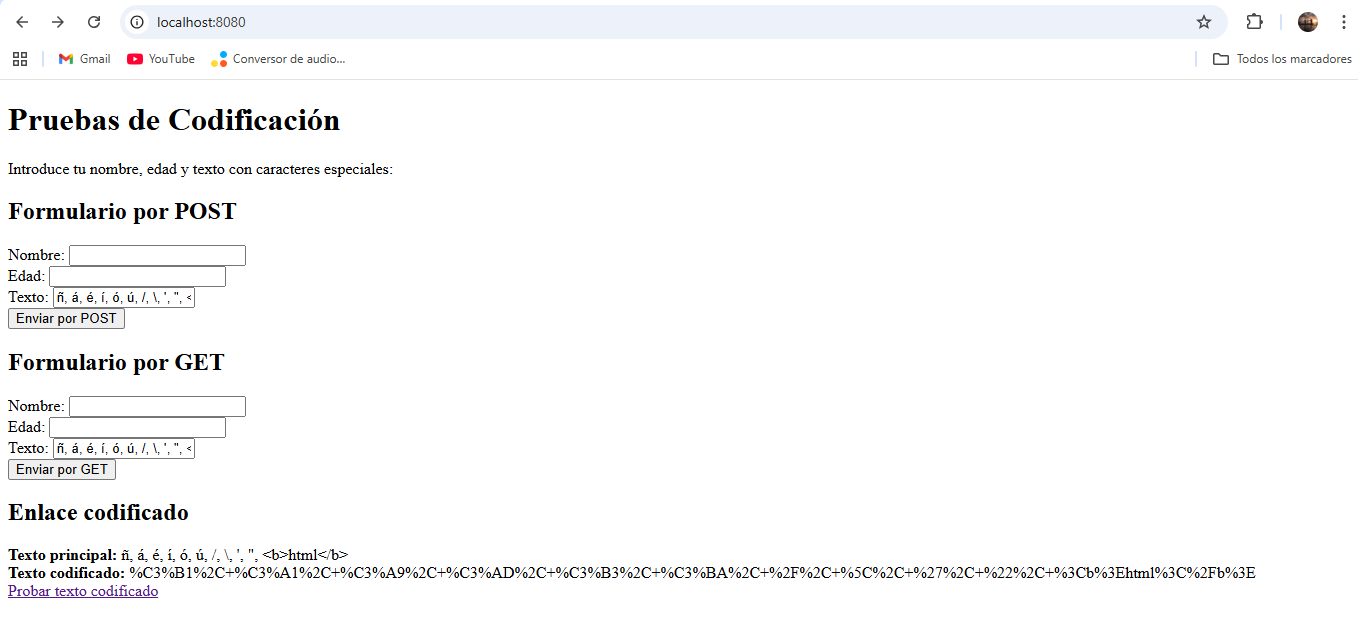


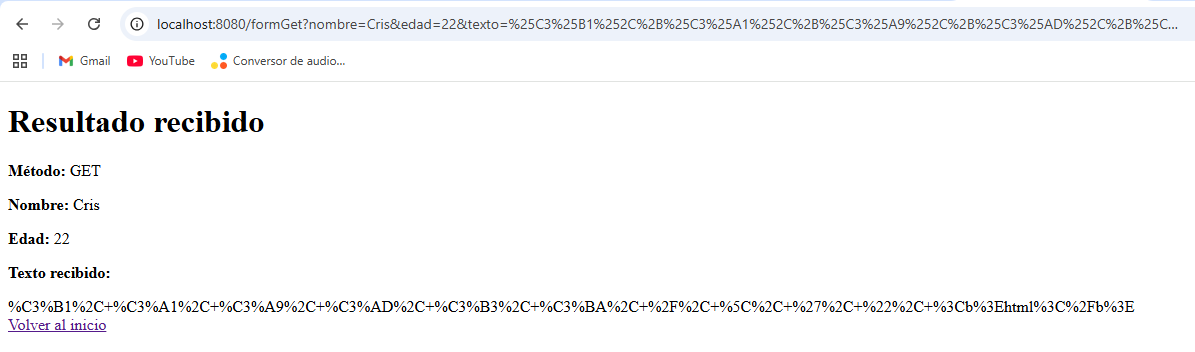
**Al usar el formulario por GET, escribimos nuestro nombre, la edad, un texto y luego hacemos clic en Enviar.**





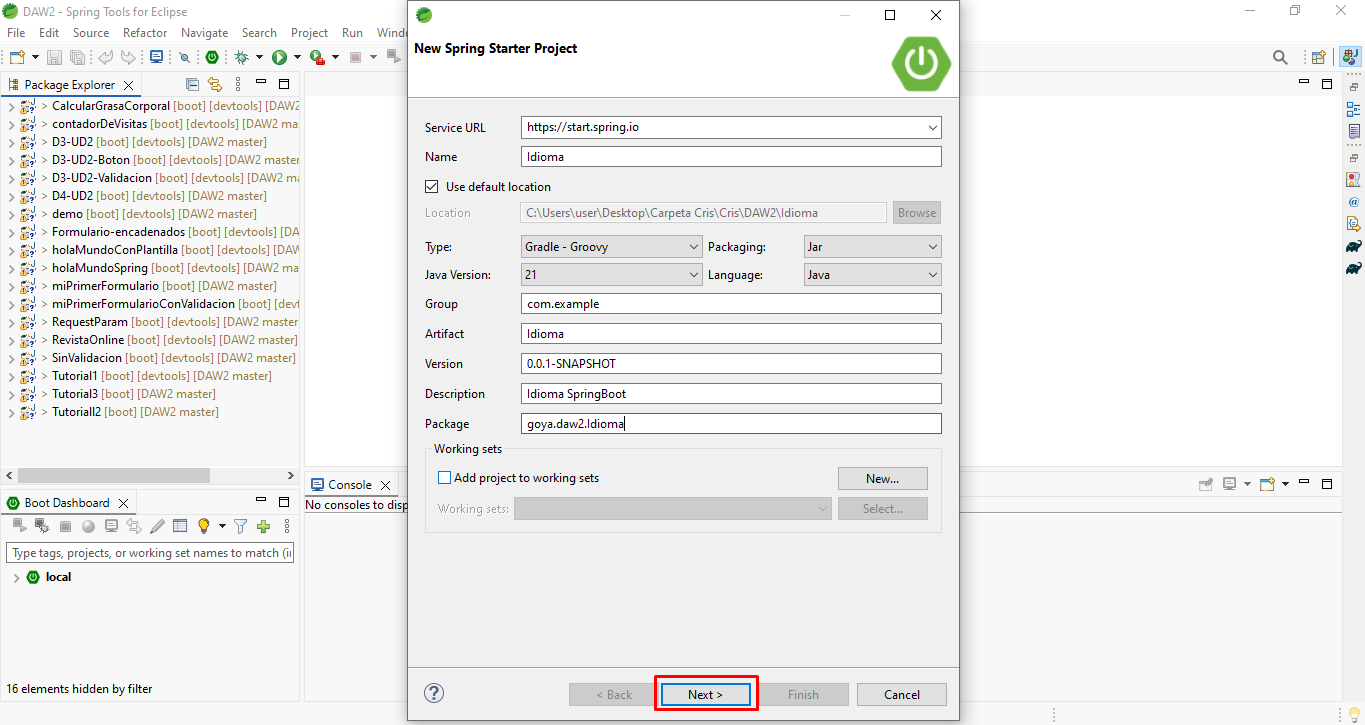
**Al usar el enlace codificado, se envían automáticamente los datos ya codificados en la URL al servidor, permitiendo ver cómo se representan y decodifican los caracteres especiales sin necesidad de rellenar el formulario.**



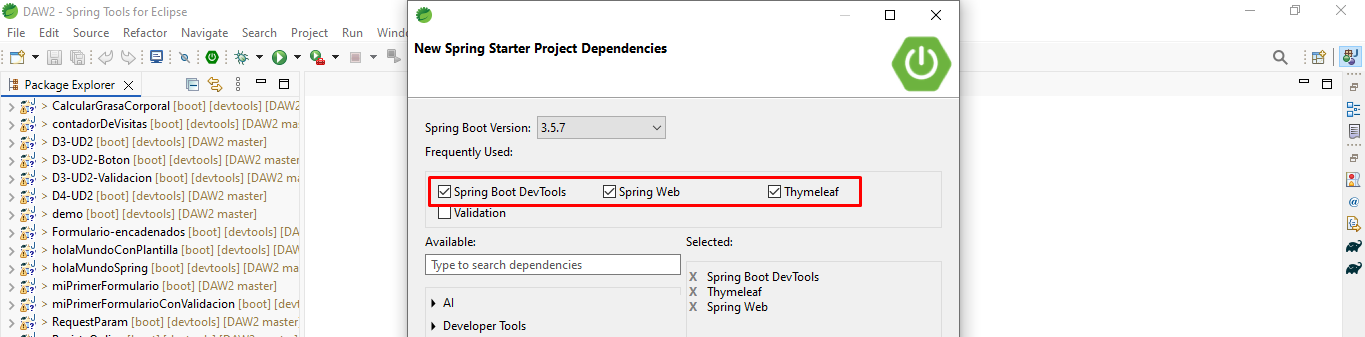


1. Realiza una prueba de contexto de tener una aplicación web mínima (cualquier de los tutoriales iniciales) pero cuyos mensajes de texto se extraigan de un fichero y cambien según el idioma. Puedes seguir estas guías: <https://aulavirtual33.educa.madrid.org/ies.goya.madrid/mod/page/view.php?id=52684>

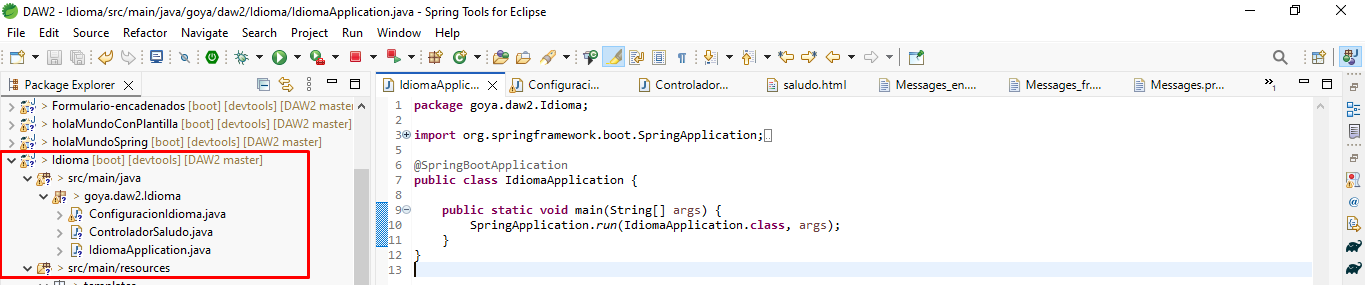
**Creamos un proyecto llamado Idioma**



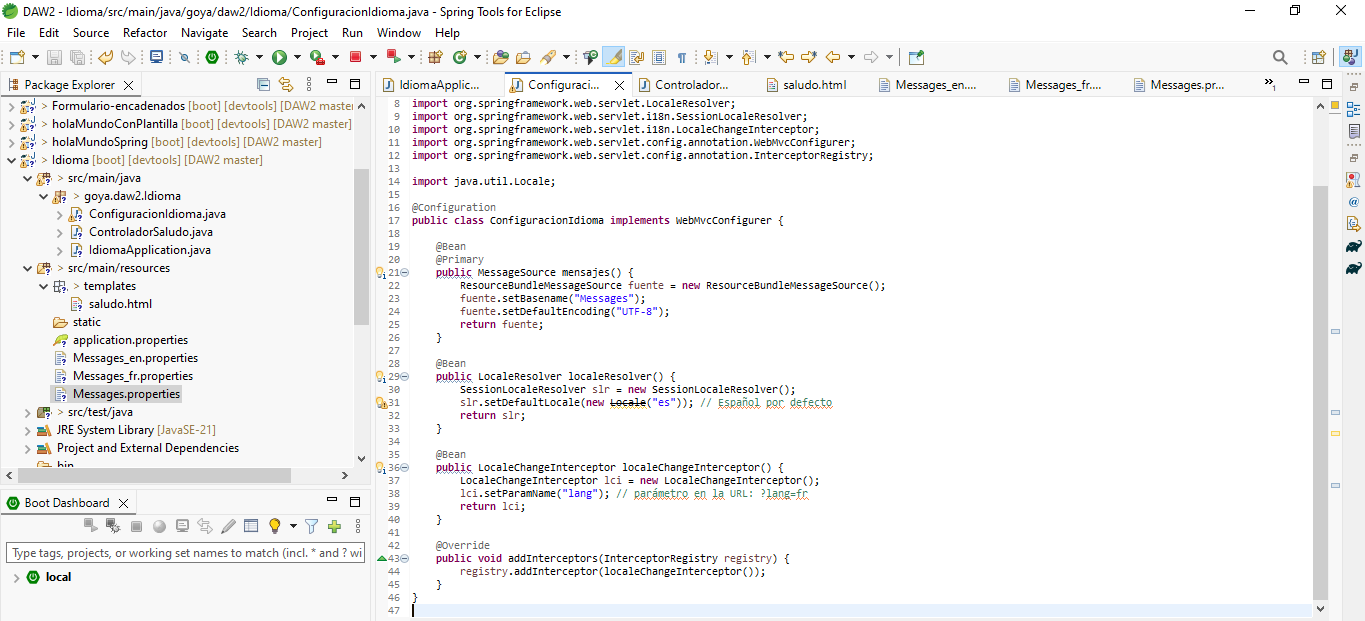
**Añadimos las dependencias: Spring Web, Thymeleaf y Spring Boot DevTools**



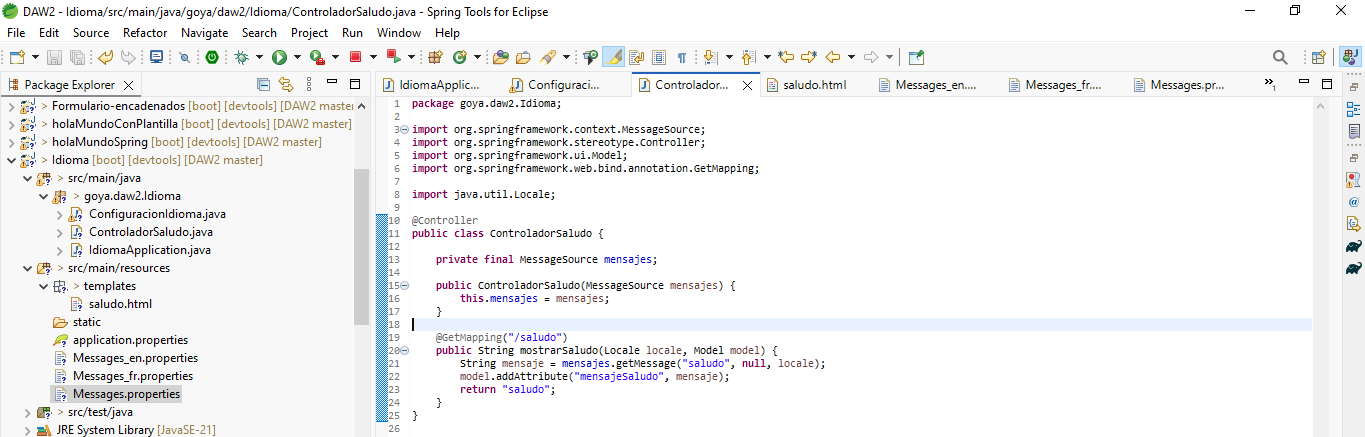
**Nos crea el proyecto y la clase por defecto**



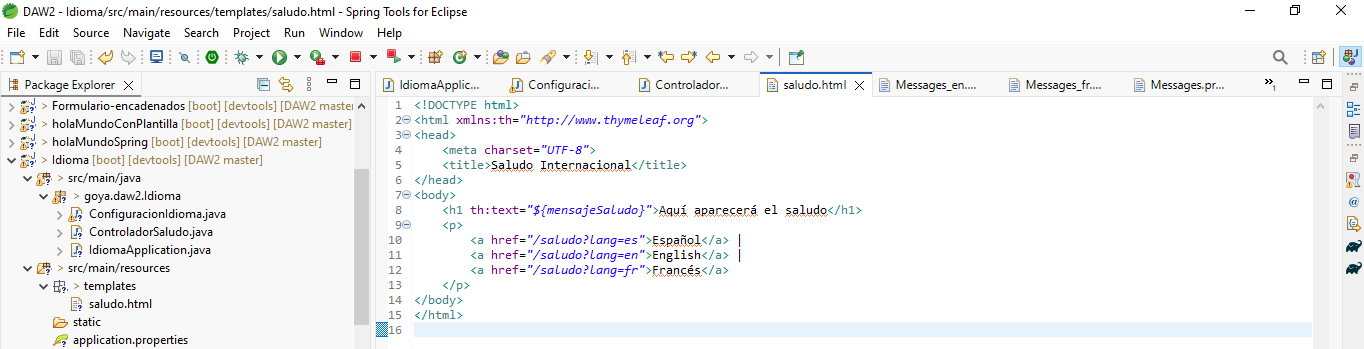
**Creamos la clase ConfiguracionIdioma, le dice a la aplicación qué idioma usar por defecto y dónde encontrar los textos traducidos, permitiendo mostrar los mensajes**



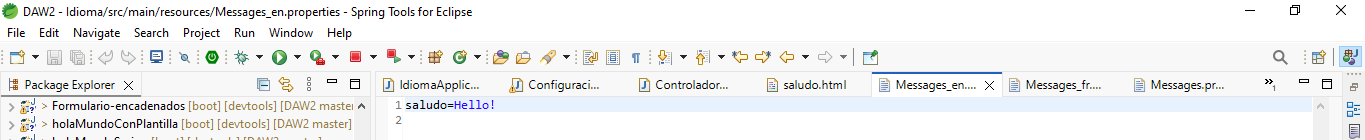
**Creamos la clase ControladorSaludo, recibe la petición de saludo, detecta el idioma del usuario, obtiene el mensaje traducido y lo envía a la vista para mostrarlo en la página.**



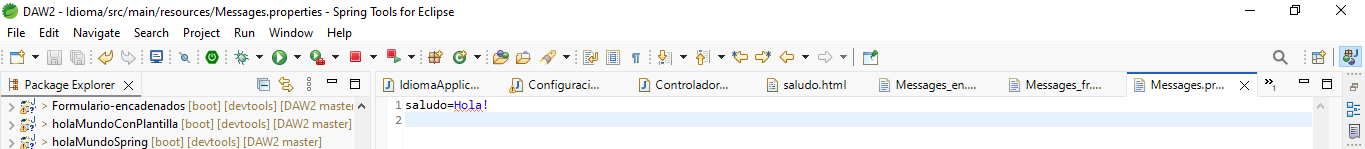
**Creamos el HTM llamado saludo.html, para mostrar en la página web el mensaje de saludo que recibe desde el ControladorSaludo**



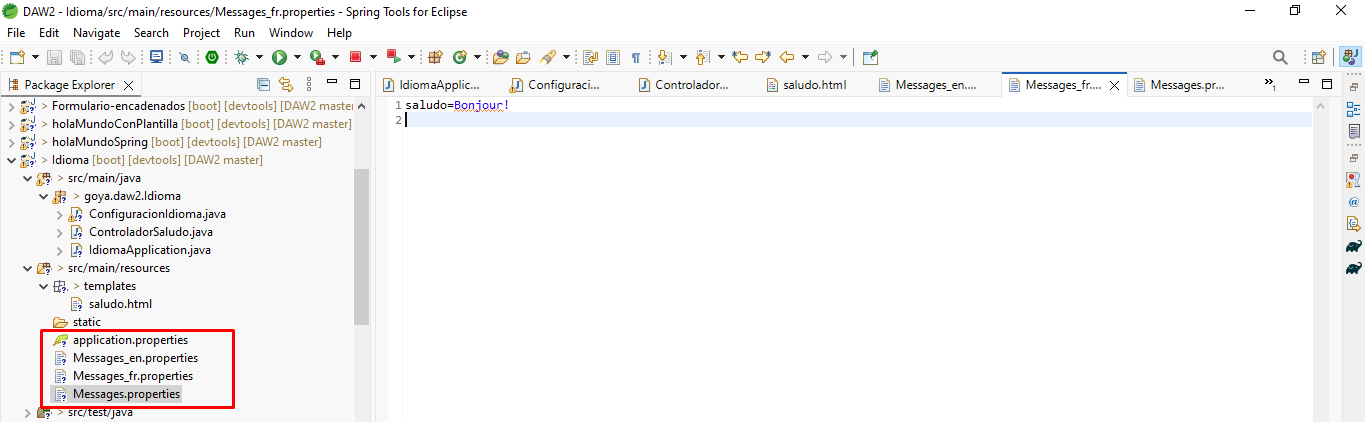
**Creamos en src/main/resources, ponemos saludo=Hello! en el archivo de mensajes para que la aplicación pueda mostrar el saludo en inglés cuando el usuario tenga configurado ese idioma.**



**Creamos en src/main/resources, ponemos saludo=Hola! en el archivo de mensajes para que la aplicación pueda mostrar el saludo en español cuando el usuario tenga configurado ese idioma.**

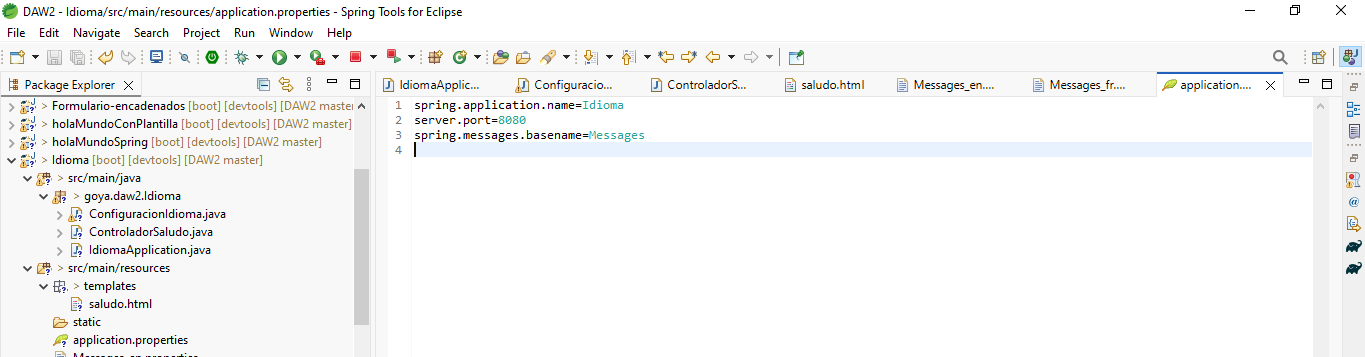


**Creamos en src/main/resources, ponemos saludo=** **Bonjour! en el archivo de mensajes para que la aplicación pueda mostrar el saludo en francés cuando el usuario tenga configurado ese idioma.**



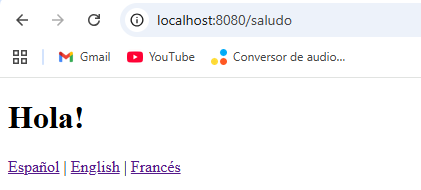
**En el archivo application.properties, sirven para configurar:**

* **spring.application.name=Idioma, útil en registros y monitorización.**
* **server.port=8080, define el puerto donde se ejecutará la aplicación.**
* **spring.messages.basename=Messages, indica el nombre de los archivos de mensajes (Messages.properties, Messages\_en.properties…) que contienen los textos traducidos.**

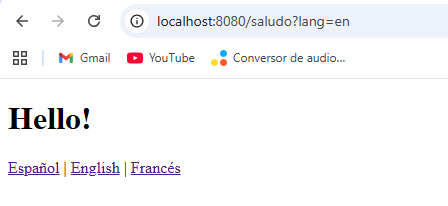


**En el navegador: http://localhost:8080/saludo**

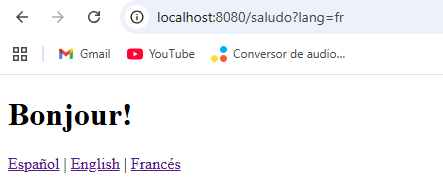
**Nos sale por defecto en español**



**En inglés**



**En francés**



Entrega: documento D4-UD2.PDF con la descripción de los pasos realizados, problemas encontrados, soluciones aportadas y conclusiones sobre la consecución de objetivos en cada apartado. El código se entregará en un enlace a cloud o git, ojo al .gitignore.

Calificación:

* Suficiente: apartado 1 bien documentados los puntos 1.1 y 1.2
* Bien: apartado 1 muy completo y bien documentado
* Excelente: apartados 1 y 2 completos y bien documenados.

**Enlace GitHub:**

**https://github.com/Cristina200317/PracticasEntornoServidor.git**