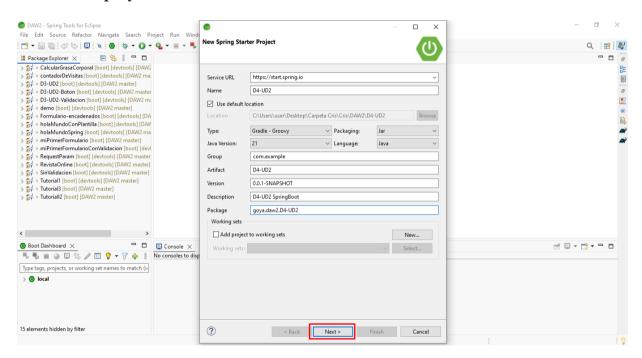
D4-UD2. Ejercicio fin UD2

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

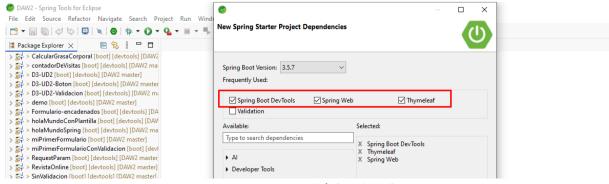


Ponemos las 3 dependencias que necesitamos:

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

Thymeleaf: para las plantillas HTML

DevTools: para cambios y reinicios del proyecto



Página 1 de 9

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

```
DAW2 - D4-UD2/src/main/java/goya/daw2/D4_UD2/D4Ud2Application.java - Spring Tools for Eclipse
                                                                                                                                                           File Edit Source Refactor Navigate Search Project Run Window Help
Q 🔡 🐯
- -
> CalcularGrasaCorporal [boot] [devtools] [DA 1 package gova.daw2.D4 UD2;
     contadorDeVisitas [boot] [devtools] [DAW2
                                                                                                                                                                  3@ import org.springframework.boot.SpringApplication;
  > D3-UD2 [boot] [devtools] [DAW2 master]
    > D3-UD2-Boton [boot] [devtools] [DAW2 ma
     > D3-UD2-Validacion [boot] [devtools] [DAW2
                                        public class D4Ud2Application {
                                                                                                                                                                  .
 ✓ 🔐 D4-UD2 [I
                                            public static void main(String[] args) {
    SpringApplication.run(D4Ud2Application.class, args);
  (0)
    y dig goya.daw2.D4_UD2

> D4Ud2Application.java
                                        }

    templates
```

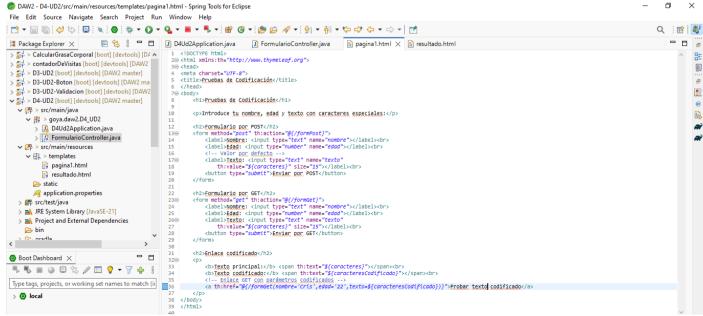
Creamos la clase FormularioController, se encarga de recibir los datos que el usuario introduce en los formularios y mostrarlos en pantalla. Su objetivo es comprobar cómo se codifican los textos con símbolos especiales cuando se envían al servidor mediante los métodos GET o POST.

```
DAW2 - D4-UD2/src/main/java/goya/daw2/D4_UD2/FormularioController.java - Spring Tools for Eclipse
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     П
  File Edit Source Refactor Navigate Search Project Run Window Help
Q 🔡 🐉
                                                                              🖹 💲 🖁 🗖 🗓 D4Ud2Application.java 📝 FormularioController.java 🗶 🖺 pagina1.html 📓 resultado.html
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            - - -
    > 🐩 > CalcularGrasaCorporal [boot] [devtools] [D# \land
                 > contadorDeVisitas [boot] [devtools] [DAW2
> D3-UD2 [boot] [devtools] [DAW2 master]
                                                                                                                                              10 public class FormularioController {
                                                                                                                                                                        // Páginal con formularios y enlace de prueba
       > D3-UD2-Boton [boot] [devtools] [DAW2 ma
                    > D3-UD2-Validacion [boot] [devtools] [DAW2
                                                                                                                                                                  <code>@GetMapping('')</code>
public <code>String</code> index(Model modelo) {
    string caracters = ^{\circ}i, ^{\circ

▼ # > D4-UD2 [boot] [devtools] [DAW2 master]

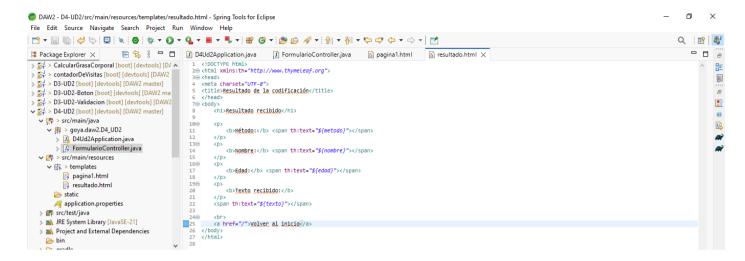
            ✓ 🚌 > src/main/java
        ・ 詩 > src/main/java
・ 詩 > goya.daw2.D4_UD2
> ほ D4Ud2Application.java
> ほ FormularioController.java
・ 母 > src/main/resources
・ 母, > templates
                                                                                                                                                                                modelo.addAttribute("caracteres", caracteres);
modelo.addAttribute("caracteresCodificado", caracteresCodificado);
                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                2
                                                                                                                                                               // Process formulario Post
@PostMapping("/formPost")
public String processPost(@RequestParam("nombre") String nombre, @RequestParam("Rdad") String edad,
@RequestParam("Edato") String texto, Model modelo) {
                     pagina1.html
resultado.html
                                                                                                                                                                             modelo.addAttribute("metodo", "POST");
modelo.addAttribute("nombre", nombre);
modelo.addAttribute("edad", edad);
modelo.addAttribute("texto", texto);
return "resultado";
         > M Project and External Dependencies
                                                                                                                                                                     // Procesa formulario GET / enlace
@GetMapping("/formGet")
                                                                                                                                                                    $ 8 0 0 0 5 / 0 0 7 + 1
 Type tags, projects, or working set names to match (ii
    > O local
```

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.



Página 2 de 9

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/

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



Pruebas de Codificación

Introduce tu nombre, edad y texto con caracteres especiales:

Formulario por POST



Resultado recibido

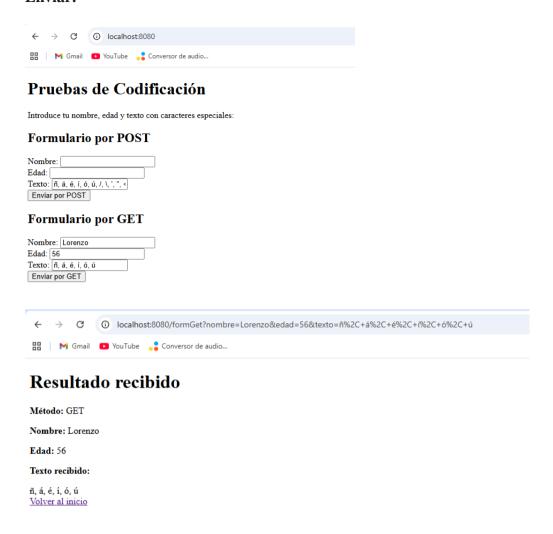
Método: POST

Nombre: Cris

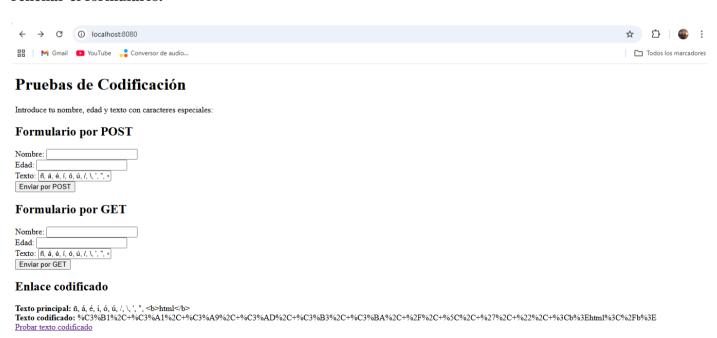
Edad: 22

Texto recibido:
ñ, á, é, í, ó, ú, /, \, ', ", html
Volver al inicio

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.





Resultado recibido

Método: GET Nombre: Cris Edad: 22

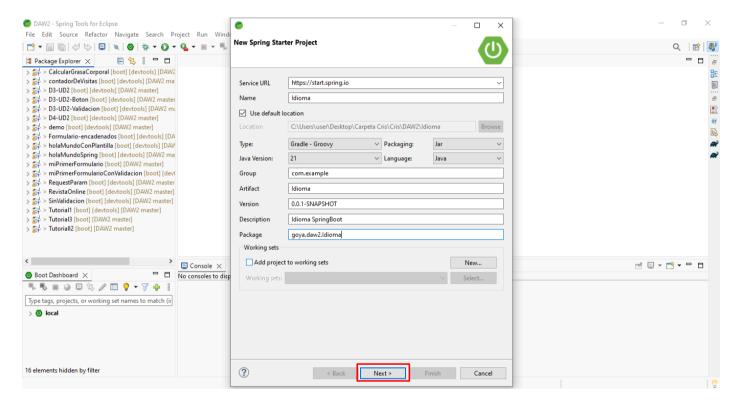
Texto recibido:

%C3%B1%2C+%C3%A1%2C+%C3%A9%2C+%C3%AD%2C+%C3%B3%2C+%C3%BA%2C+%2F%2C+%5C%2C+%27%2C+%22%2C+%3Cb%3Ehtml%3C%2Fb%3E Volver al inicio

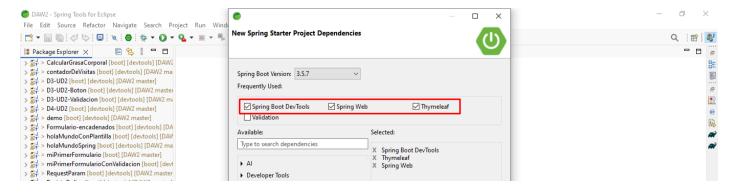
2. Realiza una prueba de contexto de tener una aplicación web mínima (cualquier de los tutoriales iniciales) pero cuyos mensajes de texto se extraigan de un fichero y cambien según el idioma. Puedes seguir estas guías:

https://aulavirtual33.educa.madrid.org/ies.gova.madrid/mod/page/view.php?id=52684

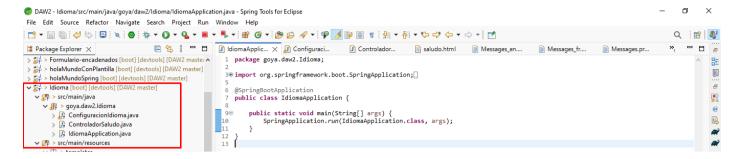
Creamos un proyecto llamado Idioma



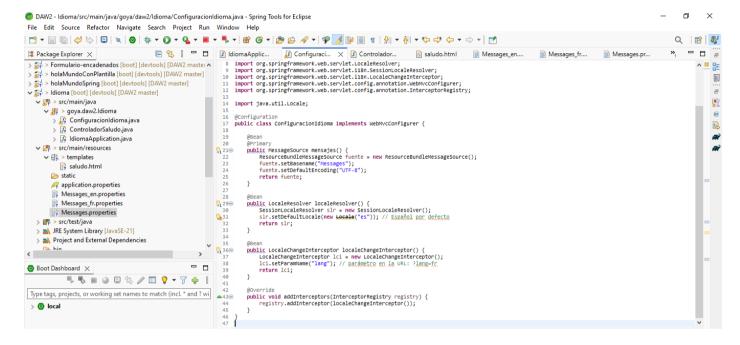
Añadimos las dependencias: Spring Web, Thymeleaf y Spring Boot DevTools



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.

```
DAW2 - Idioma/src/main/java/gova/daw2/Idioma/ControladorSaludo.java - Spring Tools for Eclipse
File Edit Source Refactor Navigate Search Project Run Window Help
Q : p | 数
☐ Package Explorer ×
                                          🖹 😤 🖁 🗖 🗓 IdiomaApplic...
                                                                                        > Formulario-encadenados [boot] [devtools] [DAW2 master A 1 package goya.daw2.Idioma;
 > 👼 > holaMundoConPlantilla [boot] [devtools] [DAW2 master]
> 👸 > holaMundoSpring [boot] [devtools] [DAW2 master]
                                                                     30 import org.springframework.context.MessageSource;
4 import org.springframework.stereotype.Controller;
5 import org.springframework.ui.Model;
6 import org.springframework.wib.oblind.annotation.GetMapping;

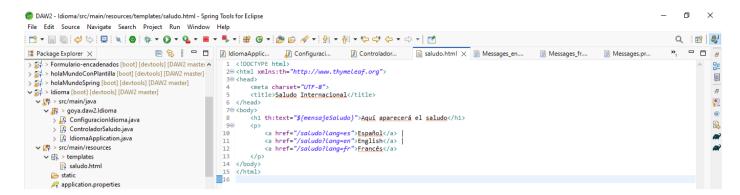
▼ $\frac{1}{28} > Idioma [boot] [devtools] [DAW2 master]

    import java.util.Locale;
                                                                 10 @Controller
11 public class ControladorSaludo {
12
13 private final MessageSource n
14
15 public ControladorSaludo(Mass
                                                                                                                                                                                                                                             >   ControladorSaludo.java
           [A] IdiomaApplication.java
                                                                                                                                                                                                                                             ai
ai
                                                                            private final MessageSource mensajes;

→ C > src/main/resources

       public ControladorSaludo(MessageSource mensajes) {
   this.mensajes = mensajes;
        geetMapping("/saludo")
public String mostrarsaludo(locale locale, Model model) {
   String messaje = mensajes.getMassage("saludo", null, locale);
   model.addattribute("mensajesaludo", mensaje);
   return "saludo";
         🚑 application.properties
         Messages_en.properties
         Messages_fr.properties
        Messages.properties
    > 📭 > src/test/java
   > M JRE System Library [JavaSE-21]
```

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



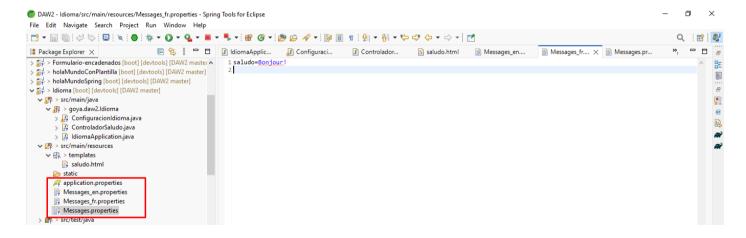
Creamos en src/main/resources, ponemos saludo=Hello! en el archivo de mensajes para que la aplicación pueda mostrar el saludo en inglés cuando el usuario tenga configurado ese idioma.



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.

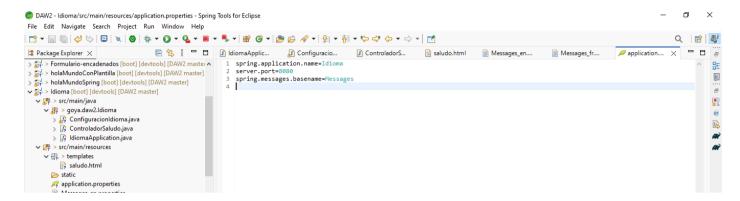


Creamos en src/main/resources, ponemos saludo= Bonjour! en el archivo de mensajes para que la aplicación pueda mostrar el saludo en francés cuando el usuario tenga configurado ese idioma.



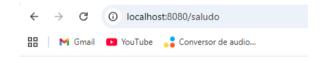
En el archivo application.properties, sirven para configurar:

- 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

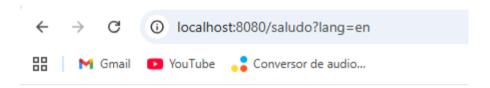
Nos sale por defecto en español



Hola!

Español | English | Francés

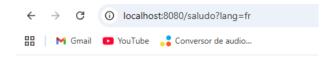
En inglés



Hello!

Español | English | Francés

En francés



Bonjour!

Español | English | 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