Contenido

[**Sección 81: Spring Framework 5: Formularios Thymeleaf y Data Binding** 3](#_Toc147216349)

[631. Creando nuevo proyecto form 3](#_Toc147216350)

[632. Actualización: para Spring Boot 2.3.0 o superior agregar dependencia validation 3 min 3](#_Toc147216351)

[633. Añadiendo la vista del formulario 3](#_Toc147216352)

[634. Método handler para recibir y procesar los datos enviados 7 min 3](#_Toc147216353)

[635. Creando la clase model del formulario 3](#_Toc147216354)

[636. Mapeando los campos del formulario al objeto model 3](#_Toc147216355)

[637. Validación del formulario usando anotaciones 3](#_Toc147216356)

[638. Añadiendo los mensajes de errores en la vista del formulario 3](#_Toc147216357)

[639. Mostrando los mensajes de errores en la vista del formulario 3](#_Toc147216358)

[640. Probando el formulario 3](#_Toc147216359)

[641. La anotación @SessionAttributes para manter los datos durante el ciclo del form 3](#_Toc147216360)

[642. La anotación @Size para rangos y @Email para validar correo electrónico 3](#_Toc147216361)

[643. Mensajes de errores personalizados usando properties 3](#_Toc147216362)

[644. Archivos properties y acentos con UTF-8 3](#_Toc147216363)

[645. Validación personalizada usando anotación @Pattern para expresiones regulares 3](#_Toc147216364)

[646. Validación personalizada usando una clase Validation 3](#_Toc147216365)

[647. Registrando una clase validador con la anotación @InitBinder 4](#_Toc147216366)

[648. Validación personalizada usando anotaciones 4](#_Toc147216367)

[649. Validación personalizada usando 4](#_Toc147216368)

[anotaciones ejemplo requerido 4](#_Toc147216369)

[650. Validación de números enteros con @Max @Min y NotNull 4](#_Toc147216370)

[651. Validación de fechas con @NotNull 4](#_Toc147216371)

[dando formato con @DateTimeFormat 4](#_Toc147216372)

[652. Validación de fechas con input type date de HTML5 y @Past y @Future 4](#_Toc147216373)

[653. Formateando fechas con @lnitBinder y registrando un CustomDateEditor 4](#_Toc147216374)

[654. Formateando fechas con @lnitBinder y registrando un CustomDateEditor Parte 2 4](#_Toc147216375)

[655. Implementando propio filtro custom property editor para convertir a mayúscula 4](#_Toc147216376)

[656. Lista select desplegable 4](#_Toc147216377)

[657. Formateando fechas en vistas thymeleaf 4](#_Toc147216378)

[658. Llenando lista select con Map 4](#_Toc147216379)

[659. Llenando lista select con objetos del la clase Pais 4](#_Toc147216380)

[660. Añadiendo componente service de Pais 4](#_Toc147216381)

[661. Añadiendo componente pais property editor 4](#_Toc147216382)

[662. Implementando checkboxes 4](#_Toc147216383)

[663. Llenando checkboxes con Map 4](#_Toc147216384)

[664. Llenando checkboxes con objetos del tipo Role 5](#_Toc147216385)

[665. Añadiendo componente role property editor 5](#_Toc147216386)

[666. Checkbox booleano true o false 5](#_Toc147216387)

[667. Radio button 5](#_Toc147216388)

[668. Input type hidden 5](#_Toc147216389)

[669. Poblando pais y roles 5](#_Toc147216390)

[670. Redirect después del POST procesar 5](#_Toc147216391)

[671. Añadiendo estilos CSS al formulario con Bootstrap 5](#_Toc147216392)

[672. Añadiendo estilos CSS Bootstrap a la plantilla resultado 5](#_Toc147216393)

[673. Cargando archivo css Bootstrap localmente 5](#_Toc147216394)

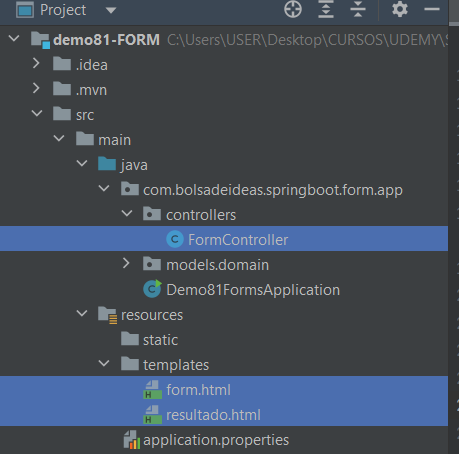
# **Sección 81: Spring Framework 5: Formularios Thymeleaf y Data Binding**

# 631. Creando nuevo proyecto form

# 632. Actualización: para Spring Boot 2.3.0 o superior agregar dependencia validation 3 min

# 633. Añadiendo la vista del formulario

# 634. Método handler para recibir y procesar los datos enviados



package com.bolsadeideas.springboot.form.app.controllers;  
  
  
import org.springframework.stereotype.Controller;  
import org.springframework.ui.Model;  
import org.springframework.web.bind.annotation.GetMapping;  
import org.springframework.web.bind.annotation.PostMapping;  
import org.springframework.web.bind.annotation.RequestParam;  
  
@Controller  
public class FormController {  
  
 @GetMapping("/form")  
 public String form(Model model){  
 model.addAttribute("titulo","Creando usuario");  
 return "form";  
 }  
  
 @PostMapping("/form")  
 public String procesar(Model model,  
 @RequestParam String username,  
 @RequestParam String password,  
 @RequestParam String email){  
  
  
 model.addAttribute("titulo","Procesando usuario");  
  
 model.addAttribute("username",username);  
 model.addAttribute("password",password);  
 model.addAttribute("email",email);  
  
 return "resultado";  
 }  
  
}

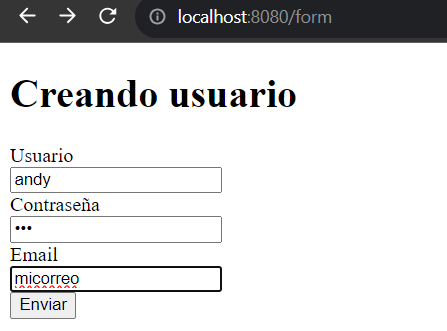
form.html

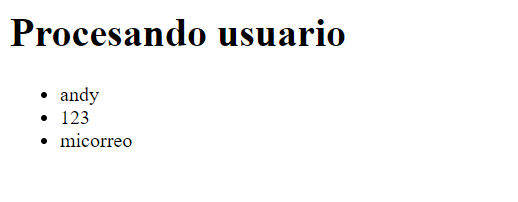
<!doctype html>  
<html lang="en" xmlns:th="http://thymeleaf.org">  
  
<head>  
 <meta charset="UTF-8">  
 <meta name="viewport"  
 content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">  
 <meta http-equiv="X-UA-Compatible" content="ie=edge">  
 <title>Form</title>  
</head>  
  
<body>  
  
<h1 th:text="${titulo}"></h1>  
  
 <form th:action="@{/form}">  
  
 <div>  
 <label for="username">Usuario </label>  
 <div>  
 <input type="text" name="username" id="username">  
 </div>  
 </div>  
  
 <div>  
 <label for="password">Contraseña </label>  
 <div>  
 <input type="password" name="password" id="password">  
 </div>  
 </div>  
  
 <div>  
 <label for="email">Email </label>  
 <div>  
 <input type="text" name="email" id="email">  
 </div>  
 </div>  
  
 <div>  
 <div>  
 <input type="submit" value="Enviar">  
 </div>  
 </div>  
  
 </form>  
  
</body>  
  
</html>

Resulado.html

<!doctype html>  
<html lang="en" xmlns:th="http://thymeleaf.org">  
  
<head>  
 <meta charset="UTF-8">  
 <meta name="viewport"  
 content="width=device-width, user-scalable=no, initial-scale=1.0, maximum-scale=1.0, minimum-scale=1.0">  
 <meta http-equiv="X-UA-Compatible" content="ie=edge">  
 <title>Form</title>  
</head>  
  
<body>  
  
<h1 th:text="${titulo}"></h1>  
  
<ul>  
  
 <li th:text="${username}"></li>  
 <li th:text="${password}"></li>  
 <li th:text="${email}"></li>  
</ul>

</body>  
  
</html>





# 635. Creando la clase model del formulario

A diferencia del anterior creamos la clase models/domain/Usuario

package com.bolsadeideas.springboot.form.app.models.domain;  
  
import lombok.Data;  
  
@Data  
public class Usuario {  
  
 //deben ser igual al name del form  
  
 private String username;  
 private String password;  
 private String email;  
  
}

Modificamos el metodo con el Post del controller (PARA ALMACENAR EL ANTERIOR COMENTAMOS Y LO VOLVIMOS A CREAR)

package com.bolsadeideas.springboot.form.app.models.domain;  
  
import lombok.Data;  
  
@Data  
public class Usuario {  
  
 //deben ser igual al name del form  
  
 private String username;  
 private String password;  
 private String email;  
  
}

FormController

instanciamos la clase Usuario y la add al model

@PostMapping("/form")  
public String procesar(Model model,  
 @RequestParam String username,  
 @RequestParam String password,  
 @RequestParam String email){  
  
 model.addAttribute("titulo","Procesando usuario");  
  
 Usuario usuario = new Usuario();  
 usuario.setUsername(username);  
 usuario.setPassword(password);  
 usuario.setEmail(email);  
  
 model.addAttribute("usuario",usuario);  
  
 return "resultado";  
}

form.html

no se modifico nada

resultado.html

solo agregamso el usuario.

<ul>  
 <!-- usuario es del cotroller -->  
 <li th:text="${usuario.username}"></li>  
 <li th:text="${usuario.password}"></li>  
 <li th:text="${usuario.email}"></li>  
</ul>

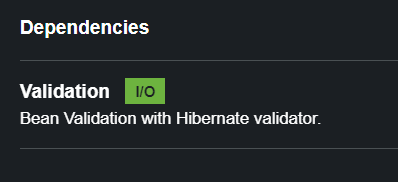
# 636. Mapeando los campos del formulario al objeto model

N el controller movemos la instancia a los parámetros del metodo

//MAPEAMOS LOS CAMPOS DEL FORMULARIO AL OBJETO MODEL  
@PostMapping("/form")  
public String procesar( Usuario usuario , Model model){  
  
 model.addAttribute("titulo","Procesando usuario");  
 model.addAttribute("usuario",usuario);  
  
 return "resultado";  
}

El resto esta igual

# 637. Validación del formulario usando anotaciones



<!-- SIRVE PARA EL @Valid-->  
 <dependency>  
 <groupId>javax.validation</groupId>  
 <artifactId>validation-api</artifactId>  
 <version>2.0.1.Final</version>  
 </dependency>

Con eso funciona el @Valid en el backend

# 638. Añadiendo los mensajes de errores en la vista del formulario

# 639. Mostrando los mensajes de errores en la vista del formulario

# 640. Probando el formulario

# 641. La anotación @SessionAttributes para manter los datos durante el ciclo del form

# 642. La anotación @Size para rangos y @Email para validar correo electrónico

# 643. Mensajes de errores personalizados usando properties

# 644. Archivos properties y acentos con UTF-8

# 645. Validación personalizada usando anotación @Pattern para expresiones regulares

# 646. Validación personalizada usando una clase Validation

# 647. Registrando una clase validador con la anotación @InitBinder

# 648. Validación personalizada usando anotaciones

# 649. Validación personalizada usando

# anotaciones ejemplo requerido

# 650. Validación de números enteros con @Max @Min y NotNull

# 651. Validación de fechas con @NotNull

# dando formato con @DateTimeFormat

# 652. Validación de fechas con input type date de HTML5 y @Past y @Future

# 653. Formateando fechas con @lnitBinder y registrando un CustomDateEditor

# 654. Formateando fechas con @lnitBinder y registrando un CustomDateEditor Parte 2

# 655. Implementando propio filtro custom property editor para convertir a mayúscula

# 656. Lista select desplegable

# 657. Formateando fechas en vistas thymeleaf

# 658. Llenando lista select con Map

# 659. Llenando lista select con objetos del la clase Pais

# 660. Añadiendo componente service de Pais

# 661. Añadiendo componente pais property editor

# 662. Implementando checkboxes

# 663. Llenando checkboxes con Map

# 664. Llenando checkboxes con objetos del tipo Role

# 665. Añadiendo componente role property editor

# 666. Checkbox booleano true o false

# 667. Radio button

# 668. Input type hidden

# 669. Poblando pais y roles

# 670. Redirect después del POST procesar

# 671. Añadiendo estilos CSS al formulario con Bootstrap

# 672. Añadiendo estilos CSS Bootstrap a la plantilla resultado

# 673. Cargando archivo css Bootstrap localmente