



UD 2: Modelo Vista Controlador

6.- Formularios: Hibernate Validator



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

Objetivos de la sesión:

- *Entender como podemos realizar **validaciones de los beans con “Hibernate Validator”** (una implementación de los estándares de validación JSR 303 y JSR 349).*
- Utilizar Hibernate Validator para **añadir nuevos datos y para modificar datos existentes.**
- ***Generar el archivo ejecutable de nuestra aplicación web y subirla a un servidor Tomcat. Para ellos aprenderemos a :***
 - Generar el archivo ejecutable de nuestra aplicación web
 - Instalar y realizar una configuración básica de un servidor Tomcat
 - Desplegar nuestra aplicación en el contenedor de aplicaciones java Tomcat.
 - Probar la aplicación: Pararla , replegarla y volverla a instalar.

UD 2: Modelo Vista Controlador

6.- Formularios: Hibernate Validator



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

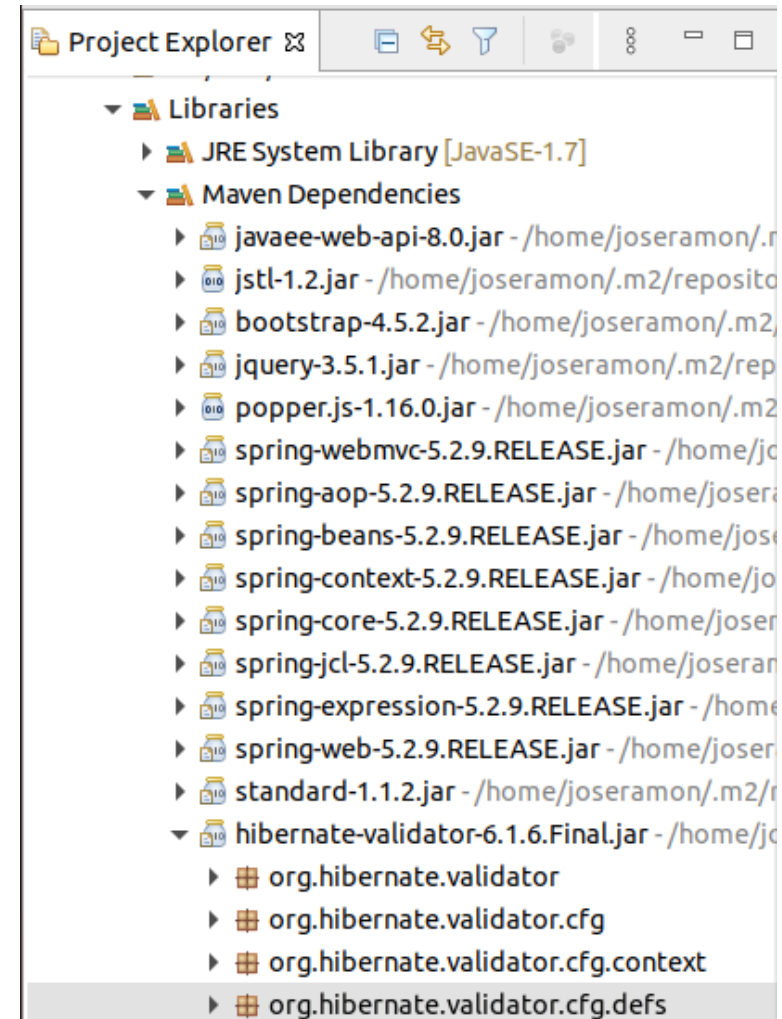
Vamos a centrarnos de lleno en la validación de formularios utilizando un **API de validación de Beans** que cumple con los estándares: **Hibernate Validator es una implementación de los estándares de validación JSR 303 y JSR 349.**

1º Hibernate Validator: En primer lugar añadiremos la dependencia:
(podemos buscar en google “hibernate validator maven”)

```

joseramon_primer_app_spring_mvc/pom.xml
21<dependency>
22    <groupId>jstl</groupId>
23    <artifactId>jstl</artifactId>
24    <version>1.2</version>
25</dependency>
26<dependency>
27    <groupId>org.webjars</groupId>
28    <artifactId>bootstrap</artifactId>
29    <version>4.5.2</version>
30</dependency>
31<dependency>
32    <groupId>org.hibernate</groupId>
33    <artifactId>hibernate-validator</artifactId>
34    <version>6.1.6.Final</version>
35</dependency>
36</dependencies>
37<build>
38    <pluginManagement>

```





UD 2: Modelo Vista Controlador

6.- Formularios: Hibernate Validator



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

... continuación **1º Hibernate Validator**:

Si **ponemos en marcha la web** veremos que nos da un error:

[INFO] Initializing Servlet 'dispatcher'

nov. 05, 2020 4:13:09 P. M. org.hibernate.validator.internal.util.Version <clinit>

INFO: HV000001: Hibernate Validator 6.1.6.Final

[WARNING] Exception encountered during context initialization - cancelling refresh attempt: [org.springframework.beans.factory.BeanCreationException](#): Error creating bean with name 'org.springframework.validation.beanvalidation.OptionalValidatorFactoryBean#0': Invocation of init method failed; nested exception is java.lang.NoClassDefFoundError: javax/el/ELManager

[ERROR] Context initialization failed

[org.springframework.beans.factory.BeanCreationException](#): Error creating bean with name 'org.springframework.validation.beanvalidation.OptionalValidatorFactoryBean#0': Invocation of init method failed; nested exception is java.lang.NoClassDefFoundError: javax/el/ELManager



¿Que nos dice y como podemos arreglarlo?



UD 2: Modelo Vista Controlador

6.- Formularios: Hibernate Validator



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

... continuación **1º Hibernate Validator**:

[INFO] Initializing Servlet 'dispatcher'

nov. 05, 2020 4:13:09 P. M. org.hibernate.validator.internal.util.Version <clinit>

INFO: HV000001: Hibernate Validator 6.1.6.Final

[WARNING] Exception encountered during context initialization - cancelling refresh attempt: [org.springframework.beans.factory.BeanCreationException](#): Error creating bean with name 'org.springframework.validation.beanvalidation.OptionalValidatorFactoryBean#0': Invocation of init method failed; nested exception is java.lang.NoClassDefFoundError: javax/el/ELManager

[ERROR] Context initialization failed

[org.springframework.beans.factory.BeanCreationException](#): Error creating bean with name 'org.springframework.validation.beanvalidation.OptionalValidatorFactoryBean#0': Invocation of init method failed; nested exception is java.lang.NoClassDefFoundError: javax/el/ELManager

No se encuentra la clase ***javax.el.ELManager (javax.servlet-api)***. Este error tiene su origen en la versión de Tomcat que estamos utilizando. Tomcat 7 utiliza “el-api” en su versión 2.2. Pero la clase ELManager fue introducido en el-api en la versión 3.0. Si arrancamos la aplicación en Tomcat 8 o superior tendremos el problema resuelto.



¿Como podemos arrancar en Tomcat 8 o superior si el plugin de maven su última versión es la 7 ?



UD 2: Modelo Vista Controlador

6.- Formularios: Hibernate Validator

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



... continuación **1º Hibernate Validator**:

Para *arrancar un Tomcat 8 o superior en Eclipse* tenemos **2 opciones**:

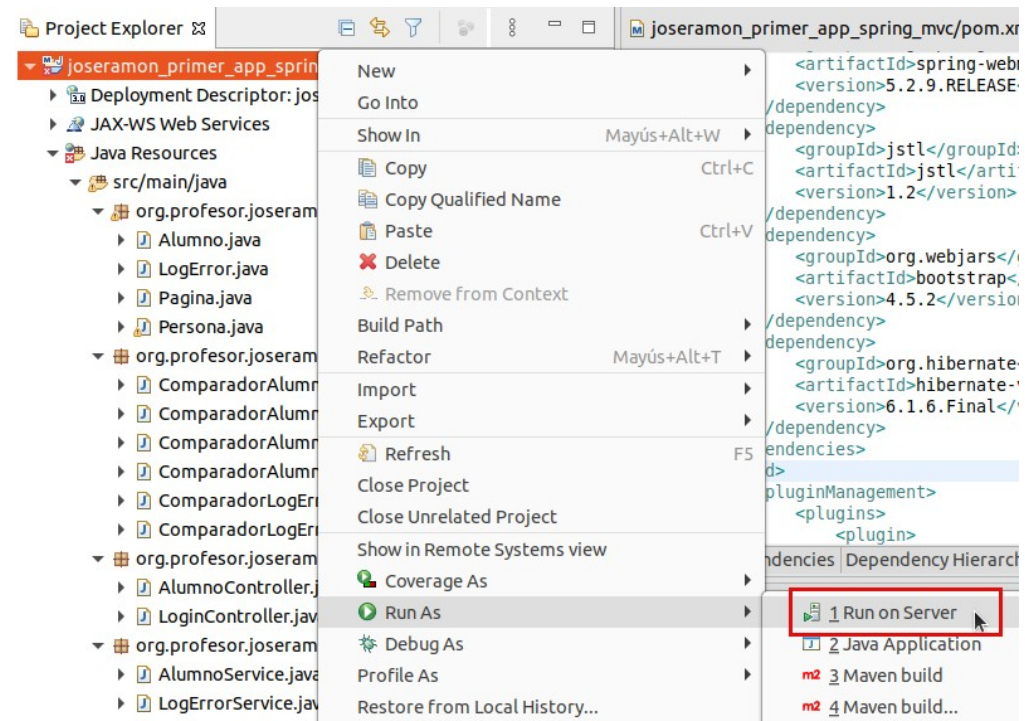
1º Podemos *enlazar el plugin de Tomcat7 con una versión posterior* instalada:

<https://stackoverflow.com/questions/41326911/maven-plugin-for-tomcat-9>

2º Podemos *arrancar con el Tomcat 9 (que instalamos en su día) directamente*.

Optamos por la **2º opción** por ser la más sencilla y **seleccionamos** “Run as\ Run on Server”

!!!Se ha acabado utilizar “Maven build” si todavía lo utilizábamos!!!



UD 2: Modelo Vista Controlador

6.- Formularios: Hibernate Validator

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



... continuación **1º Hibernate Validator**:

Seleccionamos un servidor existente y pulsamos en “Finish”:

Run On Server

Select which server to use

How do you want to select the server?

☒ Choose an existing server

☐ Manually define a new server

Select the server that you want to use:

type filter text

Server	State
localhost	
Tomcat v9.0 Server at localhost	Stopped

Apache Tomcat v9.0 supports J2EE 1.2, 1.3, 1.4, and Java EE 5, 6, 7, and 8 Web modules.

☐ Always use this server when running this project

Columns...

< Back Next > Cancel Finish



UD 2: Modelo Vista Controlador

6.- Formularios: Hibernate Validator



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

... continuación **1º Hibernate Validator**:

Si utilizabamos “Maven build” la ruta era “http://localhost:8080”, pero si ahora ejecutamos “Run on Server” la ruta principal de nuestra aplicación web nos ha cambiado: <http://localhost:8080/nombreAplicación/> .

Asegurate de adaptar las rutas en los ficheros jsp a la nueva ruta para que no falle si utilizabas “Maven build”

Si nuestra aplicación se convierte en un **fichero .war en un Tomcat en producción, la ruta contendrá el nombre de la aplicación**, por lo que incluso nos viene bien arreglar este problema para evitar el típico “En mi equipo funcionaba...”.

AYUDA: Las rutas de los jsp son respecto a servidor (http://localhost:8080), pero en los controladores empiezan con la ruta que pongamos en el dispatcher. Eso significa que en los controladores podemos empezar por “/” aunque la nueva url sea más larga.

joseramon_primer_app_spring_mvc/pom.xml Web Browser

← → ↻ ⚙ http://localhost:8080/joseramon_primer_app_spring_mvc/ ▶

Simarro Home Alumnos Modulos Errores DWES Logout

Introduzca su nombre:

Introduzca su contraseña:

Login

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

UD 2: Modelo Vista Controlador

6.- Formularios: Hibernate Validator

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



2º Validaciones: Ahora que ya contamos con el validador de Hibernate podemos incorporar 2 validaciones que hacíamos en el JSP en el Bean "Alumno.java" con anotaciones:

Project Explorer

javax.validation.constraints

AssertFalse.class
AssertTrue.class
DecimalMax.class
DecimalMin.class
Digits.class
Email.class
Future.class
FutureOrPresent.class
Max.class
Min.class
Negative.class
NegativeOrZero.class
NotBlank.class
NotEmpty.class
NotNull.class
Null.class
Past.class
PastOrPresent.class
Pattern.class
Positive.class
PositiveOrZero.class
Size.class

Alumno.java

```

1 package org.profesor.joseramon.joseramon_primer_app_spring_mvc.model;
2
3 import java.io.Serializable;
4
5 import javax.validation.constraints.Size;
6
7 public class Alumno implements Serializable, Comparable<Alumno>{
8     private static final long serialVersionUID = 1L;
9     @Size(min=9,message="El DNI debe de tener un tamaño mínimo de 9")
10    private String dni;
11    @Size(min=5,message="El nombre debe de tener un tamaño mínimo de 5")
12    private String nombre;
13    private Integer edad;
14    private String ciclo;
15    private Integer curso;
16
17    public Alumno() {
18    }
19
20    public Alumno(String dni) {
21        super();
22        this.dni = dni;
23    }
24
25    public Alumno(String dni, String nombre, Integer edad, String ciclo, Integer curso) {
26        super();
27        this.dni = dni;
28        this.nombre = nombre;
29        this.edad = edad;
30        this.ciclo = ciclo;
31        this.curso = curso;
32    }

```




3º Validar formulario:

Incorporar el tag `@Valid` al controlador:

Para que el controlador pueda realizar la validación del Bean debemos añadirle la **notación** `@Valid`:

```
AlumnoController.java
44
45 @RequestMapping(value="/add-alumno",method = RequestMethod.POST)
46 public String addAlumno(ModelMap model, @Valid Alumno alumno) {
47     String errores="";
48     paginaServicio.setPagina(pagina);
49     model.addAttribute("pagina",paginaServicio.getPagina());
50     try {
51         alumnoService.addAlumno(alumno);
52         //Para evitar pasar parámetros innecesarios
53         model.clear();
54         /* Para evitar inserciones duplicadas comentamos código y redirigimos a listar*/
55         return "redirect:list-alumno";
56     } catch (AlumnoDuplicadoException e) {
57         errores=e.toString();
58         model.addAttribute("errores",errores);
59         return "add-alumno";
60     }
61 }
62
```



UD 2: Modelo Vista Controlador

6.- Formularios: Hibernate Validator



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

... continuación **3º Validar formulario:**



¿Como validamos los resultados del formulario con Spring desde el controlador?

UD 2: Modelo Vista Controlador

6.- Formularios: Hibernate Validator



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

... continuación **3º Validar formulario:**

Para validar los resultados desde el controlador añadiremos un parámetro nuevo de tipo `BindingResult` en el método y comprobaremos dentro del método si la validación tiene errores o no para devolver la ejecución al formulario para corregirlos:

```
AlumnoController.java
44
45 @RequestMapping(value="/add-alumno",method = RequestMethod.POST)
46 public String addAlumno(ModelMap model, @Valid Alumno alumno, BindingResult validacion) {
47     if (validacion.hasErrors()) {
48         //Hay errores y debemos volver al formulario de alta
49         return "add-alumno";
50     }
51     //Si llega aquí no hay errores de validación
52     String errores="";
53     paginaServicio.setPagina(pagina);
54     model.addAttribute("pagina",paginaServicio.getPagina());
55     try {
56         alumnoService.addAlumno(alumno);
57         //Para evitar pasar parámetros innecesarios
58         model.clear();
59         /* Para evitar inserciones duplicadas comentamos código y redirigimos a listar*/
60         return "redirect:list-alumno";
61     } catch (AlumnoDuplicadoException e) {
62         errores=e.toString();
63         model.addAttribute("errores",errores);
64         return "add-alumno";
65     }
66 }
```



UD 2: Modelo Vista Controlador

6.- Formularios: Hibernate Validator



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

... continuación **3º Validar formulario:**

Si **ponemos en marcha la web** vemos que nos devuelve a la misma página para corregir el error, pero no nos muestra ningún error , ni marca en azul la sección “Alumnos”:

[Simarro](#)

[Home](#)

[Alumnos](#)

[Modulos](#)

[Errores](#)

[DWES](#)

[Logout](#)

Nuevo alumno:

Introduzca los datos del nuevo alumno:

Dni:

Nombre:

Edad:

Ciclo:

Curso:

[Añadir](#)

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



¿ Por qué ?


UD 2: Modelo Vista Controlador

6.- Formularios: Hibernate Validator

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



... continuación **3º Validar formulario:**

Si ejecutamos el servidor en modo "Debug" con  y ponemos un punto de interrupción en el return (Toggle Breakpoint) podemos ver que el modelo tiene el alumno, pero no tiene el objeto pagina. Pon la página al modelo antes de hacer el return y arreglamos ese problemilla:

AlumnoController.java

```

44
45 @RequestMapping(value="/add-alumno",method = RequestMethod.POST)
46 public String addAlumno(ModelMap model, @Valid Alumno alumno, BindingResult result) {
47     //Si llega aquí no hay errores de validación
48     if (validation.hasErrors()) {
49         //Hay errores y debemos volver al formulario de alta
50         //model.addAttribute("alumno",alumno);
51         return "add-alumno";
52     }
53     String errores="";
54     paginaServicio.setPagina(pagina);
55     model.addAttribute("pagina",paginaServicio.getPagina());
56     try {
57         alumnoService.addAlumno(alumno);
58         //Para evitar pasar parámetros innecesarios
59         model.clear();
60         /* Para evitar inserciones duplicadas comentamos código y redirigimos */
61         return "redirect:list-alumno";
62     } catch (AlumnoDuplicadoException e) {
63         errores=e.toString();
64         model.addAttribute("errores",errores);
65         return "add-alumno";
66     }
67 }
68

```

(x)= Variables
Breakpoint
Expressi

Name	Value
no method return v	
▶ this	AlumnoController (id=75)
▼ model	BindingAwareModelMap (id=183)
▼ [0]	LinkedHashMap\$Entry<K,V> (id=183)
▶ key	"nombre" (id=183)
▶ value	"joseramon" (id=184)
▼ [1]	LinkedHashMap\$Entry<K,V> (id=185)
▶ key	"alumno" (id=185)
▶ value	Alumno (id=173)
▼ [2]	LinkedHashMap\$Entry<K,V> (id=186)
▶ key	"org.springframework.validation.BindingResult.alumno" (id=186)
▶ value	BeanPropertyBindingResult (id=186)
▶ alumno	Alumno (id=173)
▶ validacion	BeanPropertyBindingResult (id=186)



4º Mostrar errores:

Para que se puedan mostrar errores debemos de añadir un nuevo tag en el fichero jsp, el tag “errors” indicando en “path” el campo que contiene el error:

```
add-alumno.jsp
1 <%@taglib uri="http://www.springframework.org/tags/form" prefix="mvc"%>
2 <%@ include file="../../jspf/header.jspf"%>
3 <%@ include file="../../jspf/menuSuperior.jspf"%>
4
5 <div class="container">
6   <h1>Nuevo alumno:</h1>
7   <p>
8     <font color="red">${errores}</font>
9   </p>
10  <p> Introduzca los datos del nuevo alumno:</p>
11
12  <mvc:form method="post" action="add-alumno" modelAttribute="alumno">
13    <div class="form-row">
14      <div class="col">
15        <mvc:label path="dni">Dni:</mvc:label>
16        <mvc:input path="dni" type="text" required="required"
17          class="form-control"/>
18        <mvc:errors path="dni" cssClass="text-warning"/>
19      </div>
```



Simarro Home Alumnos Modulos Errores DWES Logout

Nuevo alumno:

Introduzca los datos del nuevo alumno:

Dni: 555 Nombre: Nuevo Alumno

El DNI debe de tener un tamaño mínimo de 9

Edad: 18 Ciclo: DAW Curso: 2

Añadir

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

Prueba a introducir un DNI demasiado corto y comprueba que se visualiza el error.

UD 2: Modelo Vista Controlador

6.- Formularios: Hibernate Validator

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



Algunas de las anotaciones más utilizadas para validar son:

Anotación	Tipo	Función
@Length(min=,max=)	String	Comprueba si el String tiene una longitud en ese rango
@Max(value=)	Numérico o String que representa un número	Comprueba si el valor es menor o igual que el máximo
@Min(value=)	Numérico o String que representa un número	Comprueba si el valor es mayor o igual que el mínimo
@NotNull	Cualquiera	Comprueba si el valor no es nulo
@NotEmpty	Cualquiera	Comprueba si el valor no es nulo ni esta vacío
@Past	Cualquiera	Comprueba si la fecha corresponde al pasado
@Future	Cualquiera	Comprueba si la fecha corresponde al futuro
@Pattern(regex="regexp", flag=) or @Patterns({@Pattern(...)})	Cualquiera	Comprueba si el campo cumple con una expresión regular. Mirar java.util.regex.Pattern
@Range(min=,max=)	Numérico o String que representa un número	Comprueba si el tamaño del campo está entre el mínimo y el máximo
@Size(min=,max=)	String, Array, Collection o Map	Comprueba si el campo está entre el mínimo y el máximo
@AssertFalse	Cualquiera	Comprueba si la comprobación es falsa
@AssertTrue	Cualquiera	Comprueba si la comprobación es verdadera
@Valid	Objeto	Realiza la validación de dicho objeto
@Email	String	Comprueba si el valor corresponde a un email
@CreditCardNumber	String	Comprueba si el String esta bien formateado
@Digits	Numérico o String que representa un número	Comprueba si el valor es un entero o decimal correcto
@NotBlank	String	Comprueba si el String no es nulo y si le quitas los espacios comprueba que no este vacío

Es importante fijarse que una anotación funciona correctamente si se aplica a cualquiera de los tipos de datos que soporta (String , numérico,...) pero no hay garantía de que funcione bien si se aplica a otro tipo.

UD 2: Modelo Vista Controlador

6.- Formularios: Hibernate Validator

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



Sustituye todas las validaciones HTML que teníamos en el JSP por los tags “mvc:errors” correspondientes en el JSP e igual como hemos añadido la validación de tamaño del dni y el nombre en “Alumno.java”, utiliza las anotaciones que necesites para incorporar todas las validaciones en el Bean para que lo procese Hibernate Validator:

```
add-alumno.jsp
6 <h1>Nuevo alumno:</h1>
7 <p>
8 <font color="red">${errores}</font>
9 </p>
10 <p> Introduzca los datos del nuevo alumno:</p>
11
12 <mvc:form method="post" action="add-alumno" modelAttribute="alumno">
13 <div class="form-row">
14 <div class="col">
15 <mvc:label path="dni">Dni:</mvc:label>
16 <mvc:input path="dni" type="text" required="required"
17 class="form-control" minlength="9"/>
18 </div>
19 <div class="col">
20 <mvc:label path="nombre">Nombre:</mvc:label>
21 <mvc:input type="text" id="nombre" path="nombre"
22 required="required" class="form-control" minlength="5"/>
23 </div>
24 </div>
25 <div class="form-row">
26 <div class="col">
27 <mvc:label path="edad">Edad:</mvc:label>
28 <mvc:input type="number" id="edad" path="edad" required="required"
29 class="form-control" min="18" max="100"/>
30 </div>
31 <div class="col">
32 <mvc:label path="ciclo">Ciclo:</mvc:label>
33 <mvc:input type="text" id="ciclo" path="ciclo" required="required"
34 class="form-control" minlength="3"/>
35 </div>
36 <div class="col">
37 <mvc:label path="curso">Curso:</mvc:label>
38 <mvc:input type="number" id="curso" path="curso"
39 class="form-control" required="required" min="1" max="2"/>
40 </div>
41 </div>
42 <br><input type="submit" value="Añadir" class="btn btn-success">
43 </mvc:form>
44 </div>
45
46 <@ include file="..jspf/footer.jspf">
```



```
add-alumno.jsp
1 <%@taglib uri="http://www.springframework.org/tags/form" prefix="mvc"%>
2 <%@ include file="..jspf/header.jspf"%>
3 <%@ include file="..jspf/menuSuperior.jspf"%>
4
5 <div class="container">
6 <h1>Nuevo alumno:</h1>
7 <p>
8 <font color="red">${errores}</font>
9 </p>
10 <p> Introduzca los datos del nuevo alumno:</p>
11
12 <mvc:form method="post" action="add-alumno" modelAttribute="alumno">
13 <div class="form-row">
14 <div class="col">
15 <mvc:label path="dni">Dni:</mvc:label>
16 <mvc:input path="dni" type="text" required="required"
17 class="form-control"/>
18 <mvc:errors path="dni" cssClass="text-warning"/>
19 </div>
```




UD 2: Modelo Vista Controlador

6.- Formularios: Modificaciones



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

Para **modificar los datos de un alumno** seguiremos los siguientes **5 pasos**:

- 1º AlumnoService.java** : Asegurarnos que tenemos 2 métodos, uno para encontrar un alumno por su dni, y otro para poder modificar un alumno.
- 2º list-alumno.jsp**: Crear una columna extra en el listado con un botón “Modificar” para modificar cada uno de los alumnos al estilo de como lo hacemos en “Borrar”.
- 3º AlumnoController.java**: Añadir un método que sea capaz de leer un parámetro dni, recuperar dicho Alumno y enviarlo a un nuevo fichero update-alumno.jsp.
- 4º update-alumno.jsp**: Tomando como base add-alumno.jsp crear el nuevo jsp para mostrar los datos del alumno a modificar.
- 5º AlumnoController.java**: Añadir un método que sea capaz de modificar el alumno.



UD 2: Modelo Vista Controlador

6.- Formularios: Modificaciones

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



1º AlumnoService.java: Asegurarnos que tenemos 2 métodos, uno para encontrar un alumno por su dni, y otro para poder modificar un alumno:

En ***AlumnoService.java*** deberemos de tener y/o implementar 2 métodos:

- **public Alumno encontrarAlumnoPorDni(String dni):**

Deberá de devolver de la lista de alumnos el alumno con el dni pasado como parámetro

- **public void modificaAlumno(Alumno alumno):**

Como estamos guardando la información en un ArrayList el sistema más sencillo para modificar un “Alumno” (siempre que equals() funcione bien...) es que se borre el alumno existente y se añada el nuevo alumno modificado



2º `list-alumno.jsp`: Crear una columna extra en el listado con un botón “Modificar” para modificar cada uno de los alumnos al estilo de como lo hacemos en “Borrar”:

En *`list-alumno.jsp`* deberemos implementar un 2º botón en cada línea para modificar el alumno para que quede como en la imagen:

←	→	↻	🏠	🔒	localhost:8080/joseramon_primer_app_spring_mvc	⋮	🔒	☆	≡
Simarro	Home	Alumnos	Modulos	Errores	DWES	Logout			
Listado de alumnos:									
Dni	Nombre	Edad	Ciclo	Curso	Acciones				
11111111A	Jose	21	DAM	1	Modificar	Borrar			
22222222B	Pedro	32	DAW	2	Modificar	Borrar			
33333333C	Juan	23	ASIR	1	Modificar	Borrar			
Añadir alumno									



3º AlumnoController.java: Añadir un método que sea capaz de leer un parámetro dni, recuperar dicho Alumno y enviarlo a un nuevo fichero update-alumno.jsp:

En **AlumnoController.java** deberemos **implementar un nuevo método updateAlumnos** que atienda “/update-alumno” en modo GET para mostrar la vista con los datos del alumno a modificar:

4º update-alumno.jsp: Tomando como base add-alumno.jsp crear el nuevo jsp para mostrar los datos del alumno a modificar:

Realiza los cambios necesarios. Al modificar también queremos mostrar en el formulario los posibles errores utilizando la nueva notación de Spring MVC.

Simarro Home **Alumnos** Modulos Errores DWES Logout

Alumno a modificar:

Dni: Nombre:

Edad: Ciclo: Curso:



UD 2: Modelo Vista Controlador

6.- Formularios: Modificaciones



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

5º AlumnoController.java: Añadir un método que sea capaz de modificar el alumno:

En **AlumnoController.java** deberemos **implementar un nuevo método** **procesaUpdateAlumnos** que atienda “/update-alumno” en modo POST para modificar los datos del alumno. Recuerda que debemos primero validar los datos. Si hay errores mostraremos de nuevo el formulario de modificación con los errores y si no hay podremos proceder a realizar la modificación y volver a redirigir al listado.

Simarro Home **Alumnos** Modulos Errores DWES Logout

Listado de alumnos:

Dni	Nombre	Edad	Ciclo	Curso	Acciones	
22222222B	Pedro	32	DAW	2	Modificar	Borrar
33333333C	Juan	23	ASIR	1	Modificar	Borrar
11111111A	Jose Modificado	33	SMX	2	Modificar	Borrar

Añadir alumno



UD 2: Modelo Vista Controlador

7.- Tomcat



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

Aunque la práctica no ha acabado, **la parte obligatoria que se debe subir a “Aules” ya la hemos hecho.**

De **manera opcional** después de subir la práctica a “Aules” puedes continuar leyendo para saber como podemos **crear un fichero con nuestra aplicación web para subirla a un Tomcat y poder ejecutarla en producción.**

EJERCICIO:

Sigue todos los pasos de los PDF. 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 al fichero comprimido UD2_practica3_nombreAlumno.tar.gz 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 tuviésemos Windows, podemos comprimir en ZIP.



UD 2: Modelo Vista Controlador

7.- Tomcat



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

De manera opcional (aunque recomendable si tenemos Ubuntu o una máquina virtual con Ubuntu) y por curiosidad profesional vamos a **generar el archivo ejecutable de nuestra aplicación web y la vamos a subir a un Tomcat** que podría ser el de producción de nuestra empresa.

Para conseguirlo realizaremos los siguientes **4 pasos**:

1º Generar el archivo ejecutable de nuestra aplicación web

2º Instalar y realizar una configuración básica de un Tomcat

3º Desplegar nuestra aplicación en el contenedor de aplicaciones java Tomcat.

4º Pruebas: Probar la aplicación, pararla , replegarla y volverla a instalar.



¿Como generamos nuestro archivo ejecutable de la aplicación web que hemos hecho en Eclipse?



UD 2: Modelo Vista Controlador

7.- Tomcat: Empaquetar War



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

1º Generar el archivo ejecutable de nuestra aplicación web:

Para generar el archivo ejecutable de nuestra aplicación web debemos **realizar los siguientes pasos:**

1º Situarnos desde terminal en la carpeta donde tenemos el fichero **pom.xml** de nuestra aplicación web.

2º Presuponiendo que tenemos configurado “Maven” debemos de **utilizar maven para empaquetar la aplicación en un fichero .war** (el fichero que entiende Tomcat) con la instrucción : **mvn package**

```
joseramon@Notebook-PC: ~/eclipse-workspace/joseramon_primer_app_spring_mvc
joseramon@Notebook-PC:~/eclipse-workspace/joseramon_primer_app_spring_mvc$ ls
pom.xml  src  target
joseramon@Notebook-PC:~/eclipse-workspace/joseramon_primer_app_spring_mvc$ mvn package
[INFO] Scanning for projects...
[INFO]
[INFO] -----< org.profesor.joseramon:joseramon_primer_app_spring_mvc >-----
[INFO] Building joseramon_primer_app_spring_mvc 0.0.1-SNAPSHOT
[INFO] -----[ war ]-----
[WARNING] The artifact org.hibernate:hibernate-validator:jar:6.1.6.Final has been relocated to org.hibernate.validator:hibernate-validator:jar:6.1.6.Final
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ joseramon_primer_app_spring_mvc ---
[WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
[INFO] Copying 0 resource
[INFO]
[INFO] --- maven-compiler-plugin:3.8.1:compile (default-compile) @ joseramon_primer_app_spring_mvc ---
[INFO] Changes detected - recompiling the module!
[WARNING] File encoding has not been set, using platform encoding UTF-8, i.e. build is platform dependent!
```




... continuación **1º Generar el archivo ejecutable de nuestra aplicación web:**

Si no han habido errores de compilación , **el fichero “.war”** resultante lo podremos encontrar en **la carpeta “target”** de nuestro proyecto:

eclipse-workspace joseramon_...spring_mvc target			
Nombre	Tamaño	Modificación	
classes	1 elemento	mié	
generated-sources	1 elemento	Ayer	
generated-test-sources	1 elemento	Ayer	
joseramon_primer_app_spring_mvc-0.0.1-SNAPSHOT	0 elementos	11:53	
m2e-wtp	2 elementos	Ayer	
maven-archiver	1 elemento	Ayer	
maven-status	1 elemento	Ayer	
test-classes	0 elementos	mié	
joseramon_primer_app_spring_mvc-0.0.1-SNAPSHOT.war	9,4 MB	11:53	



UD 2: Modelo Vista Controlador

7.- Tomcat: Instalación en el SO



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

2º Instalar y realizar una configuración básica de un Tomcat

Presuponiendo que estamos en una máquina virtual con Ubuntu y tenemos permisos de administrador. Debemos **realizar los siguientes pasos**:

1º Actualizamos sistema: `joseramon@Notebook-PC:~$ sudo apt update`

2º Instalamos Tomcat y los principales paquetes:

```
joseramon@Notebook-PC:~$ sudo apt install -y tomcat9 tomcat9-admin
```

3º De manera **opcional** podemos instalar la documentación y ejemplos:

4º Comprobamos la instalación:

```
joseramon@Notebook-PC:~$ systemctl status tomcat9
● tomcat9.service - Apache Tomcat 9 Web Application Server
   Loaded: loaded (/lib/systemd/system/tomcat9.service; enabled; vendor preset: enabled)
   Active: active (running) since Thu 2020-11-12 17:29:14 CET; 5min ago
     Docs: https://tomcat.apache.org/tomcat-9.0-doc/index.html
   Main PID: 8987 (java)
    Tasks: 42 (limit: 19067)
   Memory: 364.7M
    CGroup: /system.slice/tomcat9.service
            └─8987 /usr/lib/jvm/default-java/bin/java -Djava.util.logging.config.file=/var/lib/tomcat9/

nov 12 17:29:17 Notebook-PC tomcat9[8987]: Starting ProtocolHandler ["http-nio-8080"]
nov 12 17:29:17 Notebook-PC tomcat9[8987]: Server startup in [2.194] milliseconds
nov 12 17:33:07 Notebook-PC tomcat9[8987]: Despliegue del descriptor de configuración [/etc/tomcat9/Catalina
nov 12 17:33:07 Notebook-PC tomcat9[8987]: The path attribute with value [/examples] in deployment de
nov 12 17:33:08 Notebook-PC tomcat9[8987]: Al menos un JAR, que se ha explorado buscando TLDs, aún no
nov 12 17:33:08 Notebook-PC tomcat9[8987]: Deployment of deployment descriptor [/etc/tomcat9/Catalina
nov 12 17:33:08 Notebook-PC tomcat9[8987]: Despliegue del descriptor de configuración [/etc/tomcat9/C
nov 12 17:33:08 Notebook-PC tomcat9[8987]: The path attribute with value [/docs] in deployment descri
nov 12 17:33:08 Notebook-PC tomcat9[8987]: Al menos un JAR, que se ha explorado buscando TLDs, aún no
nov 12 17:33:08 Notebook-PC tomcat9[8987]: Deployment of deployment descriptor [/etc/tomcat9/Catalina
lines 1-20/20 (END)
```

UD 2: Modelo Vista Controlador

7.- Tomcat: Instalación en el SO

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



... continuación **2º Instalar y realizar una configuración básica de un Tomcat**
5º Configurar el puerto del Tomcat:

```
joseramon@Notebook-PC:~$ sudo nano /etc/tomcat9/server.xml
```

Cambiamos el puerto 8080 por el **8099**, guardamos y salimos.

```

joseramon@Notebook-PC: ~
GNU nano 4.8 /etc/tomcat9/server.xml
<!--The connectors can use a shared executor, you can define one or more na>
<!--
<Executor name="tomcatThreadPool" namePrefix="catalina-exec-"
      maxThreads="150" minSpareThreads="4"/>
-->

<!-- A "Connector" represents an endpoint by which requests are received
and responses are returned. Documentation at :
Java HTTP Connector: /docs/config/http.html
Java AJP  Connector: /docs/config/ajp.html
APR (HTTP/AJP) Connector: /docs/apr.html
Define a non-SSL/TLS HTTP/1.1 Connector on port 8080
-->
<Connector port="8099" protocol="HTTP/1.1"
      connectionTimeout="20000"
      redirectPort="8443" />
<!-- A "Connector" using the shared thread pool-->
<!--
<Connector executor="tomcatThreadPool"
      [ 171 líneas leídas ]
^G Ver ayuda ^O Guardar ^W Buscar ^K Cortar Tex ^J Justificar ^C Posición
^X Salir ^R Leer fich. ^\ Reemplazar ^U Pegar ^T Ortografía ^ Ir a línea

```



... continuación **2º Instalar y realizar una configuración básica de un Tomcat**
5º Configurar los usuarios del Tomcat:

El gestor de aplicaciones web requiere usuarios con el **rol manager-gui**, mientras que el gestor de la máquina virtual requiere el **rol admin-gui**. Vamos a crear un usuario con ambos roles:

```
joseramon@Notebook-PC:~$ sudo nano /etc/tomcat9/tomcat-users.xml
```

```

joseramon@Notebook-PC: ~
GNU nano 4.8 /etc/tomcat9/tomcat-users.xml Modificado
section below since they are intended for use with the examples web
application.
-->
<!--
NOTE: The sample user and role entries below are intended for use with the
examples web application. They are wrapped in a comment and thus are ignored
when reading this file. If you wish to configure these users for use with the
examples web application, do not forget to remove the <!-- .. --> that surrounds
them. You will also need to set the passwords to something appropriate.
-->
<!--
<role rolename="tomcat"/>
<role rolename="role1"/>
<user username="tomcat" password="<must-be-changed>" roles="tomcat"/>
<user username="both" password="<must-be-changed>" roles="tomcat,role1"/>
<user username="role1" password="<must-be-changed>" roles="role1"/>
-->
<user username="AquiNombre" password="AquiContraseña" roles="admin-gui,manager-gui"/>
</tomcat-users>

^G Ver ayuda ^O Guardar ^W Buscar ^K Cortar Text ^J Justificar ^C Posición
^X Salir ^R Leer fich. ^_ Reemplazar ^U Pegar ^T Ortografía ^_ Ir a línea
  
```


UD 2: Modelo Vista Controlador

7.- Tomcat: Instalación en el SO

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



... continuación **2º Instalar y realizar una configuración básica de un Tomcat**
6º Reseteamos el servicio para aplicar los cambios:

```
joseramon@Notebook-PC:~$ sudo systemctl restart tomcat9
```

7º Ahora ya deberíamos de poder **acceder a Tomcat** en <http://localhost:8099/> :

The screenshot shows a web browser window with the address bar set to `localhost:8099`. The browser's tab bar shows several open tabs: 'Aplicaciones', 'ITACA', 'Simarro', 'Gmail', 'Kahoot!', 'Webex GV', and three folders named 'tmp', 'Simarro', and 'srv'. The main content area displays the Tomcat 9 default home page. It starts with the heading 'It works !' followed by a congratulatory message. Below this, it provides the default Tomcat home page location: `/var/lib/tomcat9/webapps/ROOT/index.html`. It also mentions the installation paths for `CATALINA_HOME` and `CATALINA_BASE`. A section titled 'You might consider installing the following packages, if you haven't already done so:' lists three packages: **tomcat9-docs**, **tomcat9-examples**, and **tomcat9-admin**, each with a brief description and a link to its documentation or examples. A final note states that the manager webapp is restricted to users with the role 'manager-gui' and the host-manager webapp is restricted to users with the role 'admin-gui'.

It works !

If you're seeing this page via a web browser, it means you've setup Tomcat successfully. Congratulations!

This is the default Tomcat home page. It can be found on the local filesystem at: `/var/lib/tomcat9/webapps/ROOT/index.html`

Tomcat veterans might be pleased to learn that this system instance of Tomcat is installed with `CATALINA_HOME` in `/usr/share/tomcat9` and `CATALINA_BASE` in `/var/lib/tomcat9`, following the rules from `/usr/share/doc/tomcat9-common/RUNNING.txt.gz`.

You might consider installing the following packages, if you haven't already done so:

tomcat9-docs: This package installs a web application that allows to browse the Tomcat 9 documentation locally. Once installed, you can access it by clicking [here](#).

tomcat9-examples: This package installs a web application that allows to access the Tomcat 9 Servlet and JSP examples. Once installed, you can access it by clicking [here](#).

tomcat9-admin: This package installs two web applications that can help managing this Tomcat instance. Once installed, you can access the [manager webapp](#) and the [host-manager webapp](#).

NOTE: For security reasons, using the manager webapp is restricted to users with role "manager-gui". The host-manager webapp is restricted to users with role "admin-gui". Users are defined in `/etc/tomcat9/tomcat-users.xml`.



UD 2: Modelo Vista Controlador

7.- Tomcat: Desplegar

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



3º Desplegar nuestra aplicación en el contenedor de aplicaciones java Tomcat:

Para desplegar nuestra aplicación pulsaremos en el enlace “**manager webapp**” o también podemos ir a la url directa ***http://localhost:8099/manager/html*** :

En dicha sección podremos ver las aplicaciones actualmente instaladas y sus rutas.

The screenshot shows the Tomcat Manager Web Application interface in a web browser. The browser address bar shows the URL `localhost:8099/manager/html`. The page features the Tomcat logo (a yellow cat) and the Apache Software Foundation logo. The main heading is "Gestor de Aplicaciones Web de Tomcat". Below this, there is a message box showing "Mensaje: OK". The interface is divided into several sections: "Gestor" with links for "Listar Aplicaciones", "Ayuda HTML de Gestor", "Ayuda de Gestor", and "Estado de Servidor"; "Aplicaciones" which is a table listing installed applications; and "Desplegar" which provides options to deploy a directory or WAR file.

Ruta	Versión	Nombre a Mostrar	Ejecutándose	Sesiones	Comandos
/	Ninguno especificado		true	0	Arrancar Parar Recargar Replegar Expirar sesiones sin trabajar ≥ 30 minutos
/docs	Ninguno especificado	Tomcat Documentation	true	0	Arrancar Parar Recargar Replegar Expirar sesiones sin trabajar ≥ 30 minutos
/examples	Ninguno especificado	Servlet and JSP Examples	true	0	Arrancar Parar Recargar Replegar Expirar sesiones sin trabajar ≥ 30 minutos
/host-manager	Ninguno especificado	Tomcat Host Manager Application	true	0	Arrancar Parar Recargar Replegar Expirar sesiones sin trabajar ≥ 30 minutos
/manager	Ninguno especificado	Tomcat Manager Application	true	1	Arrancar Parar Recargar Replegar Expirar sesiones sin trabajar ≥ 30 minutos

Desplegar
Desplegar directorio o archivo WAR localizado en servidor

Trayectoria de Contexto (opcional):



UD 2: Modelo Vista Controlador

7.- Tomcat: Desplegar



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

... continuación **3º Desplegar nuestra aplicación en el contenedor de aplicaciones java Tomcat:**

Para **desplegar nuestra aplicación** debemos bajar hasta la sección **“Archivo WAR a desplegar”** y escoger el archivo que hemos generado desde maven con el botón **“Seleccionar archivo”**:

localhost:8099/manager/html

Aplicaciones ITACA Simarro Gmail Kahoot! Webex GV tmp Simarro srv prg

Expirar sesiones sin trabajar < 30 minutos

Desplegar

Desplegar directorio o archivo WAR localizado en servidor

Trayectoria de Contexto (opcional):

Version (for parallel deployment):

URL de archivo de Configuración XML:

URL de WAR o Directorio:

Desplegar

Archivo WAR a desplegar

Seleccione archivo WAR a cargar Ningún archivo seleccionado

Configuration

Re-read TLS configuration files

TLS host name (optional)

Diagnósticos

Revisa a ver si una aplicación web ha causado fallos de memoria al parar, recargar o replegarse.

<input type="button" value="Halla fallos de memoria"/>	Este chequeo de diagnóstico disparará una colección completa de basura. Utilízalo con extremo cuidado en sistemas en producción.
--	--

TLS connector configuration diagnostics

<input type="button" value="Cifrados"/>	List the configured TLS virtual hosts and the ciphers for each.
<input type="button" value="Certificates"/>	Lista los virtual hosts configurados con TLS y la cadena de certificado para cada uno de ellos.
<input type="button" value="Trusted Certificates"/>	List the configured TLS virtual hosts and the trusted certificates for each.



UD 2: Modelo Vista Controlador

7.- Tomcat: Desplegar

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



... continuación **3º Desplegar** nuestra aplicación en el contenedor de aplicaciones **java Tomcat:**

Pulsamos en “**Desplegar**” y si todo ha ido bien ya vemos nuestra aplicación en el gestor de aplicaciones:



Gestor de Aplicaciones Web de Tomcat

Mensaje:	OK
----------	----

Gestor			
Listar Aplicaciones	Ayuda HTML de Gestor	Ayuda de Gestor	Estado de Servidor

Aplicaciones					
Ruta	Versión	Nombre a Mostrar	Ejecutándose	Sesiones	Comandos
/	Ninguno especificado		true	0	Arrancar Parar Recargar Replegar Expirar sesiones sin trabajar ≥ 30 minutos
/docs	Ninguno especificado	Tomcat Documentation	true	0	Arrancar Parar Recargar Replegar Expirar sesiones sin trabajar ≥ 30 minutos
/examples	Ninguno especificado	Servlet and JSP Examples	true	0	Arrancar Parar Recargar Replegar Expirar sesiones sin trabajar ≥ 30 minutos
/host-manager	Ninguno especificado	Tomcat Host Manager Application	true	0	Arrancar Parar Recargar Replegar Expirar sesiones sin trabajar ≥ 30 minutos
/joseramon_primer_app_spring_mvc-0.0.1-SNAPSHOT	Ninguno especificado	Lista de alumnos	true	0	Arrancar Parar Recargar Replegar Expirar sesiones sin trabajar ≥ 30 minutos
/manager	Ninguno especificado	Tomcat Manager Application	true	1	Arrancar Parar Recargar Replegar Expirar sesiones sin trabajar ≥ 30 minutos



UD 2: Modelo Vista Controlador

7.- Tomcat: Pruebas



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

4º Pruebas: Probar la aplicación, pararla , replegarla y volverla a instalar.
En primer lugar probamos nuestra aplicación:

← → ↻ ⓘ localhost:8099/joseramon_primer_app_spring_mvc-0.0.1-SNAPSHOT/list-alumno ☆ 👤 ⋮

Aplicaciones ITACA Simarro Gmail Kahoot! Webex GV tmp »

Simarro Home **Alumnos** Modulos Errores DWES Logout

Listado de alumnos:

Dni	Nombre	Edad	Ciclo	Curso	Acciones	
33333333C	Juan	23	ASIR	1	Modificar	Borrar
22222222B	Pedro2	32	DAW	2	Modificar	Borrar
44444444c	Nuevo Alumno	18	DAW	2	Modificar	Borrar

Añadir alumno



UD 2: Modelo Vista Controlador

7.- Tomcat: Pruebas



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

... continuación 4º Pruebas: Probar la aplicación, pararla , replegarla y volverla a instalar.

Arrancar	Parar	Recargar	Replegar
Expirar sesiones sin trabajar ≥ 30 minutos			

Prueba a **parar tu aplicación** pulsando en “**Parar**”, y comprueba que ya no va.

Prueba a volverla a **poner en marcha** con “**Arrancar**”.

Si queremos actualizarla con el fichero WAR pulsaremos en “**Recargar**”. Recargar sirve para cuando volvemos a subir una versión actualizada de nuestra app (desplegandola) que se llama igual poder decirle al Tomcat que actualice el WAR.

Prueba a **borrar nuestra aplicación web** del Tomcat pulsando en “**Replegar**”.



UD 2: Modelo Vista Controlador

7.- Tomcat: Pruebas



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

... continuación **4º Pruebas: Probar la aplicación, pararla , replegarla y volverla a instalar.**

La ruta que utiliza Tomcat a partir de la cual funciona nuestra aplicación web es el nombre del WAR.

Vamos a cambiarle la ruta: **Quita tu aplicación del Tomcat, renombra el fichero WAR a “nombreAlumno_spring_mvc” y vuélvela a desplegarla en Tomcat.**

OJO: Si en los ficheros JSP has puesto rutas absolutas a una carpeta en concreto no te irá...

← → ↻ ① localhost:8099/joseramon_spring_mvc/ 🔑 ☆ 👤 ⋮

📁 Aplicaciones 📁 ITACA 📁 Simarro 📁 Gmail 📁 Kahoot! 📁 Webex GV 📁 tmp 📁 Simarro 📁 srv 📁 prg »

Simarro Home Alumnos Modulos Errores DWES Logout

Introduzca su nombre:

Introduzca su contraseña:

Login

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