Acabamos de ver como usar Java equals y hashcode

Spring MVC Bean Validation es una de las características más utilizadas de Spring MVC y nos permite apoyarnos en la JSR 303 (Bean Validation) para validar la información de los objetos que estemos introduciendo con un formulario. Para ello deberemos incluir en el proyecto de Maven las dependencias de Spring MVC así como las de Hibernate Validator que implementa la especificación JSR 303.

```
&amp;lt;dependencies&amp;gt;

&amp;lt;dependency&amp;gt;
 &amp;lt;groupId&amp;gt;org.springframework&amp;lt;/groupId&amp;gt;
 &amp;lt;artifactId&amp;gt;spring-webmvc&amp;lt;/artifactId&amp;gt;
 &amp;lt;version&amp;gt;4.1.7.RELEASE&amp;lt;/version&amp;gt;
 &amp;lt;/dependency&amp;gt;
 &amp;lt;dependency&amp;gt;
 &amp;lt;groupId&amp;gt;javax.servlet&amp;lt;/groupId&amp;gt;
 &amp;lt;artifactId&amp;gt;jstl&amp;lt;/artifactId&amp;gt;
 &amp;lt;version&amp;gt;1.2&amp;lt;/version&amp;gt;
 &amp;lt;scope&amp;gt;provided&amp;lt;/scope&amp;gt;
 &amp;lt;/dependency&amp;gt;
 &amp;lt;dependency&amp;gt;
 &amp;lt;groupId&amp;gt;javax.servlet&amp;lt;/groupId&amp;gt;
 &amp;lt;artifactId&amp;gt;servlet-api&amp;lt;/artifactId&amp;gt;
 &amp;lt;version&amp;gt;2.5&amp;lt;/version&amp;gt;
 &amp;lt;scope&amp;gt;provided&amp;lt;/scope&amp;gt;
 &amp;lt;/dependency&amp;gt;
 &amp;lt;dependency&amp;gt;
 &amp;lt;groupId&amp;gt;org.hibernate&amp;lt;/groupId&amp;gt;
```

```
 &amp;lt;artifactId&amp;gt;hibernate-
validator&amp;lt;/artifactId&amp;gt;
 &amp;lt;version&amp;gt;5.1.3.Final&amp;lt;/version&amp;gt;
 &amp;lt;/dependency&amp;gt;
 &amp;lt;/dependencies&amp;gt;
```

El siguiente paso será configurar Spring para que el fichero de Application-Context.xml soporte anotaciones a nivel de controladores.

```
&amp;lt;?xml version="1.0" encoding="UTF-8"?&amp;gt;
&amp;lt;beans xmlns="http://www.springframework.org/schema/beans"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:context="http://www.springframework.org/schema/context"
 xmlns:mvc="http://www.springframework.org/schema/mvc"
xmlns:p="http://www.springframework.org/schema/p"
 xsi:schemaLocation="http://www.springframework.org/schema/mvc
http://www.springframework.org/schema/mvc/spring-mvc-3.2.xsd
 http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans.xsd
 http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/spring-context-3.2.xsd"&
amp;gt;

 &amp;lt;mvc:annotation-driven /&amp;gt;
 &amp;lt;context:component-scan base-
package="com.arquitecturajava.controller" /&amp;gt;
 &amp;lt;bean id="viewResolver"
class="org.springframework.web.servlet.view.InternalResourceViewResolv
```

```
er"&amp;gt;
 &amp;lt;property name="prefix" value="/WEB-INF/jsp/" /&amp;gt;
 &amp;lt;property name="suffix" value=".jsp" /&amp;gt;
 &amp;lt;/bean&amp;gt;

&amp;lt;/beans&amp;gt;
```

Realizadas estas dos primeras operaciones construiremos un formulario que nos permite introducir el nombre y la edad de una Persona.

```
&amp;lt;%@ page language="java" contentType="text/html;
charset=ISO-8859-1"
 pageEncoding="ISO-8859-1"%&amp;gt;
&amp;lt;%@taglib prefix="form"
 uri="http://www.springframework.org/tags/form" %&amp;gt;
&amp;lt;!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd"&amp;gt;
&amp;lt;html&amp;gt;
&amp;lt;head&amp;gt;
&amp;lt;meta http-equiv="Content-Type" content="text/html;
charset=ISO-8859-1"&amp;gt;
&amp;lt;title&amp;gt;Insert title here&amp;lt;/title&amp;gt;
&amp;lt;/head&amp;gt;
&amp;lt;body&amp;gt;
&amp;lt;form:form modelAttribute="persona" action="verPersona.do"
method="post"&amp;gt;
Nombre:&amp;lt;form:input id="nombre" path="nombre" /&amp;gt;
```

```
Edad:&amp;lt;form:input id="edad" path="edad" /&amp;gt;
&amp;lt;form:button value="enviar"
&amp;gt;enviar&amp;lt;/form:button&amp;gt;

&amp;lt;form:errors path="edad" /&amp;gt;

&amp;lt;/form:form&amp;gt;
&amp;lt;/body&amp;gt;
&amp;lt;/html&amp;gt;
```

Spring MVC Bean Validation (Business Object)

Realizados estos primeros pasos debemos implementar en la clase de negocio “Persona” las anotaciones de Bean Validation que permitan validar el campo edad.

```
package com.arquitecturajava.negocio;

import javax.validation.constraints.Min;
import javax.validation.constraints.Size;

public class Persona {

 private String nombre;

 @Min(0)
 private int edad;

 public int getEdad() {
 return edad;
```

```
 }

 public void setEdad(int edad) {
 this.edad = edad;
 }

 public String getNombre() {
 return nombre;
 }

 public void setNombre(String nombre) {
 this.nombre = nombre;
 }

}
```

Como se puede observar se ha utilizado la anotación @Min para asegurar que el valor mínimo del campo edad es cero.

Spring MVC Bean Validation (Controller)

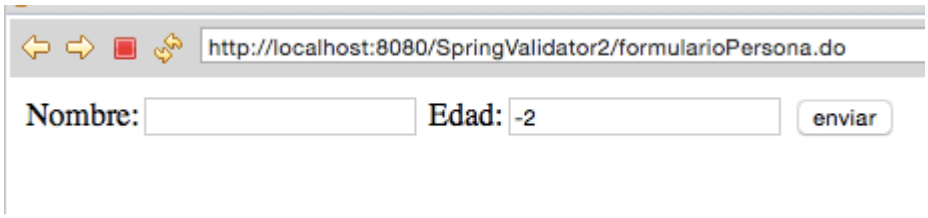
Por último queda ver como se implementa el Controlador.

```
package com.arquitecturajava.controller;

import javax.validation.Valid;

import org.springframework.stereotype.Controller;
```

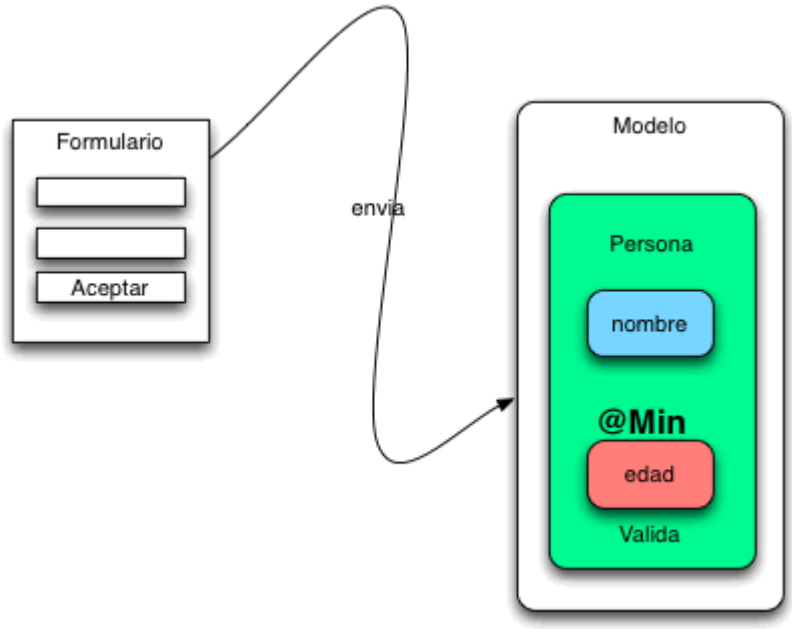
```
import org.springframework.ui.Model;
import org.springframework.validation.BindingResult;
import org.springframework.web.bind.annotation.ModelAttribute;
import org.springframework.web.bind.annotation.RequestMapping;

import com.arquitecturajava.negocio.Persona;

@Controller
public class PersonaController {

@RequestMapping(value ="/verPersona")
public String enviar(@Valid Persona persona, BindingResult
bindingResult) {
if (bindingResult.hasErrors()) {

return "formularioPersona";
}else {
return "verPersona";
}

}
@RequestMapping(value="/formularioPersona")
public String formularioPersona(@ModelAttribute Persona persona) {

return "formularioPersona";
}
}
```

En este caso el método enviar incluye una anotación @Valid que se encarga de comprobar si las validaciones de la clase Persona se cumplen o no. En el caso de que se cumplan pasamos

a una página de destino que muestra los datos de la Persona. En caso contrario retornamos el control a la página original si la edad es negativa.



Con estos datos Spring MVC creará un objeto de tipo Persona y le pasará la información del formulario. Las anotaciones de Bean Validation comprobarán si el objeto es válido.



Al no serlo nos enviará de vuelta al formulario mostrándonos un mensaje de error que clarifique lo que ha sucedido.

De esta forma habremos integrado Bean Validation con Spring MVC. Una de las grandes ventajas de Bean Validation es que únicamente implementamos las validaciones en un lugar.

Otros artículos relacionados :

Spring MVC Configuración

JSF vs Spring MVC