Cada día se usa más Spring MVC como framework de capa de presentación. Con los años se ha pasado de un modelo MVC con fuerte uso de ficheros XML a un modelo en el que priman las anotaciones. Una de las anotaciones más habituales es Spring MVC @ModelAttribute que nos permite realizar un binding de los datos que tenemos en un formulario de Spring con la capa de backend.

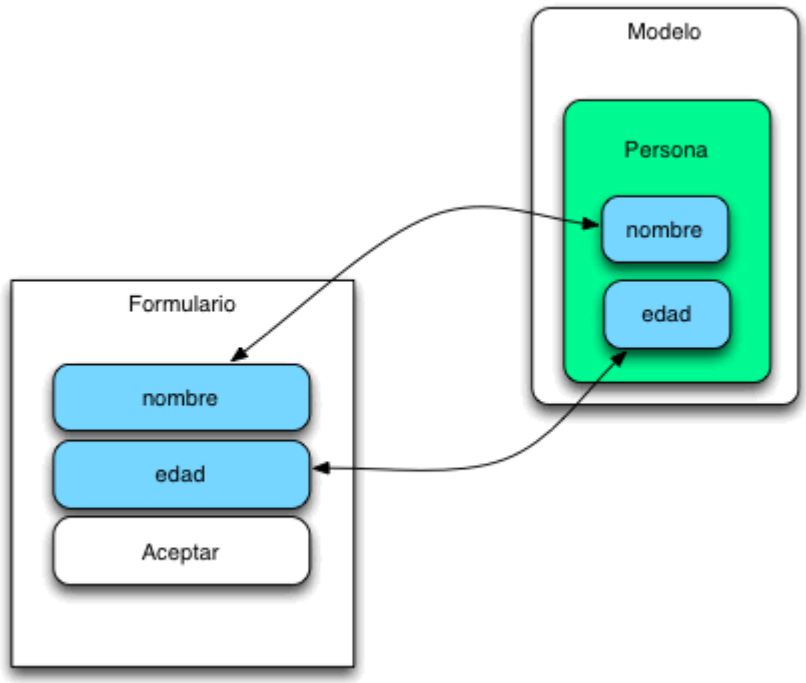
@ModelAttribute

Supongamos que tenemos un Controller en Spring MVC con dos métodos que mapean URLs (formularioPersona y verPersona)

```
package com.arquitecturajava.controller;

import org.springframework.stereotype.Controller;
import org.springframework.ui.Model;
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(Persona persona) {
 return "verPersona";
 }

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

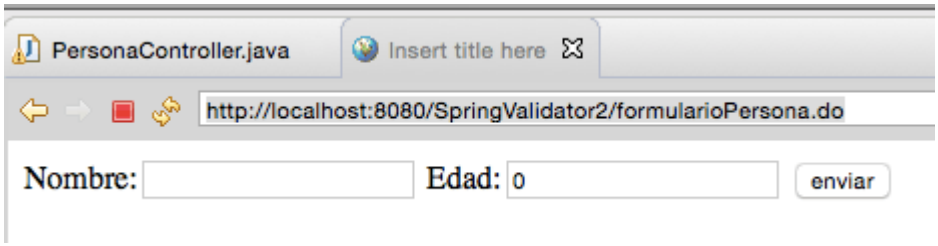
Como se puede observar la url de /formularioPersona asignará un objeto Persona al Modelo utilizando la anotación @ModelAttribute. Este objeto Persona será inicializado con los valores por defecto. El siguiente paso será ligar la Persona a los campos del formulario.



Para realizar esta operación se utilizan las librerías de etiquetas de Spring MVC.

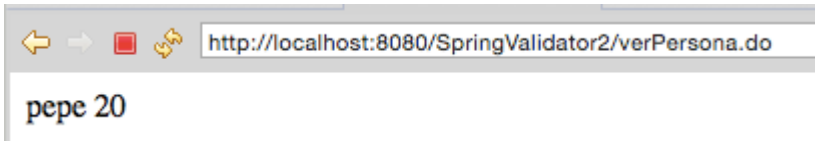
```
&amp;lt;%@ page language="java" contentType="text/html;
charset=ISO-8859-1"
 pageEncoding="ISO-8859-1"%&amp;gt;
&amp;lt;%@taglib prefix="form"
 uri="http://www.springframework.org/tags/form" %&amp;gt;
&amp;lt;!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd"&amp;gt;
&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:form&amp;gt;
&amp;lt;/body&amp;gt;
&amp;lt;/html&amp;gt;
```

Como se puede observar es suficiente con definir el atributo “modelAttribute” en el formulario y asignar el objeto Persona. Al solicitar la URL el framework nos mostrará el formulario con los datos a rellenar.



De esta forma quedan completamente ligadas ambas partes. Al rellenar el formulario y pulsar sobre el botón de enviar se ejecutará el mapeo de "/verPersona".Este mapeo como se puede observar inyecta un objeto Persona al método "enviar" . Este objeto contendrá los datos que hemos rellenado en el formulario.

Recibida esta información los datos serán pasados a la página "verPersona" que nos mostrará la información



Para mostrar la información correctamente usaremos Expression Language:

```
&amp;lt;%@ page language="java" contentType="text/html;
charset=ISO-8859-1"
pageEncoding="ISO-8859-1"%&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;
${persona.nombre}
${persona.edad}
&amp;lt;/body&amp;gt;
&amp;lt;/html&amp;gt;
```

Spring aporta muchas facilidades a la hora de trabajar con el clásico modelo MVC

Otros artículos relacionados:

Spring MVC Configuración

Spring MVC Anotaciones