Data Binding en Spring MVC

- **Data Binding** es el mecanismo mediante el cual Spring MVC **extrae dinámicamente** los datos de entrada del usuario y los asigna a objetos de **Modelo** de nuestra aplicación.
  - ✓ Un ejemplo de **Data Binding** en Spring MVC sería almacenar los datos de un formulario HTML a un objeto de Modelo (AUTOMÁTICAMENTE).
    - **AUTOMÁTICAMENTE** significa no hacerlo manualmente, por ejemplo:
      - **String name = request.getParameter("nombre")** → Como se hace comúnmente utilizando el API de Servlets.
      - **@RequestParam("nombre") String name** → Como lo hacemos en Spring MVC.

- Cuando utilizamos **Data Binding** en Spring MVC, la conversión de los tipos de datos se hace automáticamente.
  - ✓ Los parámetros de una petición HTTP son enviados como texto simple (String).
    - Spring MVC convierte los parámetros de la petición HTTP de tipo String, al tipo de dato según nuestra clase de modelo (int, Date, double, etc).

- La validación de los datos de entrada también se hace automáticamente.
  - ✓ Se puede crear una validación de datos personalizada.

Ejemplo de Data Binding

VIEW

```
<form action="/noticias/save" method="post">

  <input type="text" name="titulo" />


  <select name="estatus" >
      <option value="Activa">Activa</option>
      <option value="Inactiva">Inactiva</option>
  </select>


  <textarea name="detalle"></textarea>

 <button type="submit">Guardar</button>
</form>
```

MODEL

```
public class Noticia {

  private int id;

  private String titulo;

  private Date fecha;

  private String estatus;

  private String detalle;

  // getters / setters
}
```

CONTROLLER

```
@PostMapping("/noticias/save")
public String store(Noticia noticia){
    serviceNoticias.guardar(noticia);
    return "someView";
}
```

Al enviar el formulario (POST):

1. La petición la recibe el controlador.
2. Spring MVC extraé cada parámetro de la petición.
3. (Data Binding). Se asigna cada parámetro de la petición a cada propiedad del Modelo, pero solo si el nombre de un parámetro de la petición (input del formulario) coincide con el de una propiedad de nuestro objeto de modelo.

BindingResult – Control de errores en Data Binding

- Cuando Spring MVC realiza Data Binding pueden surgir errores. Por ejemplo:
  - ✓ Omitir campos requeridos.
  - ✓ Errores de conversión. Por ejemplo en el atributo duración (private int duracion) el usuario ingresa un valor alfanumérico.
  - ✓ Errores de formato. Por ejemplo el usuario ingresa la fecha 31-02-2017.
- Para revisar posibles errores después del Data Binding agregar un parámetro de tipo BindingResult **INMEDIATAMENTE** después de la clase de Modelo.Ejemplo:

```
@PostMapping(value = "/save")
public String guardar(Pelicula pelicula,BindingResult result) {
    System.out.println("Guardando: " + pelicula);
    return "peliculas/formPelicula";
}
```

- Despues de agregar el parámetro BindingResult, el error HTTP 400 (BAD REQUEST) ya no se muestra.

BindingResult – Verificar si hay errores

- Verificar si hay errores:

```
@PostMapping(value = "/save")
public String guardar(Pelicula pelicula,BindingResult result) {

    if (result.hasErrors()) {
        return "peliculas/formPelicula";
    }

    System.out.println("Guardando: " + pelicula);
    return "peliculas/formPelicula";
}
```

- Desplegar errores [consola]

```
for (ObjectError error: result.getAllErrors()){
   System.out.println(error.getDefaultMessage());
}
```

BindingResult – Desplegar errores en la vista

- Agregar el taglib de spring.

  <%@taglib uri="http://www.springframework.org/tags" prefix="spring" %>

- Utilizar el tag <spring:hasBindErrors> para desplegar los posibles errores que ocurrieron al realizarse el Data Binding.

```
<spring:hasBindErrors name="pelicula">
  <div class='alert alert-danger' role='alert'>
    Por favor corrija los siguientes errores:
    <ul>
      <c:forEach var="error" items="${errors.allErrors}">
         <li><spring:message message="${error}" /></li>
      </c:forEach>
    </ul>
  </div>
</spring:hasBindErrors>
```

Anotación @InitBinder – Perzonalizar Data Binding

```
Error
Failed to convert property value of type java.lang.String to required type java.util.Date for property fechaEstreno; nested exception is org.springframework.core.convert.ConversionFailedException: Failed to convert from type [java.lang.String] to type [java.util.Date] for value 21-07-2017; nested exception is java.lang.IllegalArgumentException
```

- El error anterior significa que Spring no puede convertir el valor del parámetro de la petición "21-07-2017" a un tipo Date.
  - ✓ La causa es que por Default Spring espera la fecha en el formato según este configurada en el sistema operativo donde este la aplicación.
- La anotación **@InitBinder** permite crear métodos para configurar el Data Binding directamente en el controlador.
- Los métodos marcados con la anotación @InitBinder no regresan ningún valor.
  - ✓ Normalmente son declarados como void.
- Comúnmente reciben un párametro de tipo WebDataBinder.
- El siguiente ejemplo muestra como utilizar @InitBinder para configurar CustomDateEditor para todas las propiedades de tipo java.util.Date

```
@InitBinder
public void initBinder(WebDataBinder binder) {
    SimpleDateFormat dateFormat = new SimpleDateFormat("dd-MM-yyyy");
    binder.registerCustomEditor(Date.class, new CustomDateEditor(dateFormat, false));
}
```

Redirect y Flash Attributes

- **Los atributos flash** proporcionan una forma de almacenar atributos para poder ser usados en otra petición diferente.
- Comúnmente son usados cuando hacemos una **redirección** utilizando el patrón Post/Redirect/Get.
  - ✓ Ejemplo: **return "redirect:/peliculas/index";**
- Los atributos flash son almacenados temporalmente antes de hacer el redirect **(tipícamente en la sesión)** para tenerlos disponibles después del redirect. Despúes del Redirect son eliminados de la sesión automáticamente.

Ejemplo (Controlador)

```
@PostMapping("/save")
public String guardar(Pelicula pelicula, BindingResult result, RedirectAttributes attributes) {
    if (result.hasErrors()) {
      System.out.println("Existieron errores");
      return "peliculas/formPelicula";
    }
   // Procesar objeto de modelo
   attributes.addFlashAttribute("msg", "Registro Guardado");
   return "redirect:/peliculas/index";
}
```

Ejemplo (archivo JSP)

```
<div class='alert alert-success' role="alert">${ msg }</div>
```

Registro Guardado