JAVA SERVER FACES

VALIDADORES Y CONVERTIDORES

Eric Gustavo Coronel Castillo
gcoronelc.blogspot.com
gcoronelc@gmail.com

Temas

- Mensajes de error
- Convertidores
- Validación

MENSAJES DE ERROR

- Mostrar todos los mensajes de error:

```
<h:messages style="color: red" />
```

- Mensaje de error de una componente:

```
<h:outputText value="Dirección:" />
<h:inputText id="direc"
             value="#{cliente.direccion}"
             required="true" />
<h:message for="direc" style="color: red" />
```

MENSAJES DE ERROR

- Generación de un mensaje de error:

```
FacesMessage msg;
msg  = new FacesMessage(FacesMessage.SEVERITY_ERROR,
   "Error en el proceso.",
   "La cuenta no tiene saldo suficiente");
FacesContext.getCurrentInstance().addMessage(null, msg);
```

CONVERTIDORES

- Una de los muchos mecanismos que nos proporciona JSF de forma automática y casi transparente para el desarrollador es el de la conversión.

- Es decir, convertir los campos introducidos en un formulario, objetos tipo String, al tipo de objeto Java adecuado. Y de la misma forma cuando se pinta la pagina necesitamos convertir a String el valor correspondiente de los objetos de nuestra aplicación.

- Se ejecutan antes de las validaciones.

CONVERTIDORES

- Un convertidor se utiliza para dar formato a un dato que se debe mostrar al usuario.
  - Por ejemplo, para mostrar una fecha se puede utilizar un convertidor de formato de fecha a un formato mas común para el usuario, como "DD/MM/YYYY".

- También se puede usar un convertidor en combinación con un control de entrada.
  - La entrada del usuario debe estar en el formato especificado por el convertidor. Si el formato de la entrada no coincide en el formato del convertidor se puede lanzar una excepción en el convertidor que se muestra al usuario. El objeto asociado no se actualiza hasta que se corrige el error.

CONVERTIDORES

- Conversores Estándares

BigDecimalConverter
NumberConverter
ShortConverter
CharacterConverter
DoubleConverter
DateTimeConverter

BigIntegerConverter
IntegerConverter
ByteConverter
FloatConverter
BooleanConverter

CONVERTIDORES

Ejemplo 01

En este caso el valor del control es String y la propiedad empleado.nombre tambien es de tipo String.

En este caso el valor del control es de tipo String yla propiedad empleado.salario es de tipo double, por lo tanto se debe hacer una conversión.

En este la propiedad empleado.activo es de tipo boolean, también de realizar una conversión.

```
<h:panelGrid columns="3">

   <h:outputLabel value="Nombre:"/>
   <h:inputText id="nombre" value="#{empleado.nombre}"
     required="true" requiredMessage="Nombre es obligatorio." >
   </h:inputText>
   <h:message for="nombre" errorStyle="color:red"/>

   <h:outputLabel value="Salario:"/>
   <h:inputText id="salario" label="Salario" value="#{empleado.salario}"
     required="true" requiredMessage="Salario es obligatorio.">
     <f:convertNumber type="number" pattern="###,##0.00" minFractionDigits="2" />
   </h:inputText>
   <h:message for="salario" errorStyle="color:red"/>

   <h:outputLabel value="Activo:"/>
   <h:selectBooleanCheckbox id="activo" value="#{empleado.activo}"/>
   <h:outputText />

   <h:commandButton action="#{empleado.doProcesar()}" value="Procesar"/>
</h:panelGrid >
```

CONVERTIDORES

Ejemplo 02

En este caso la fecha se debe ingresar en formato dd-mm-yyyy para que el convertidor lo pueda reconocer.

En este caso se utiliza el convertidor para mostrar un fecha con un formato full.

```
<h:panelGrid columns="3">

  <h:outputLabel value="Fecha de Ingreso (dd-mm-yyyy)"/>
  <h:inputText id="fechaIngreso" value="#{beanDatos.fechaIngreso}"
    required="true" requiredMessage="Debe ingresar la fecha."
    label="Fecha de Ingreso">
    <f:convertDateTime pattern="dd-MM-yyyy" type="date" />
  </h:inputText>
  <h:message for="fechaIngreso" errorStyle="color:red"/>

  <h:outputLabel value="Fecha de Ingressada:"/>
  <h:outputText value="#{beanDatos.otraFecha}">
    <f:convertDateTime type="date" dateStyle="full" />
  </h:outputText>
  <h:outputText />

  <h:commandButton action="#{beanDatos.doProcesar()}" value="Procesar"/>
</h:panelGrid>
```

CONVERTIDORES

Ejemplo 02

Tenemos el managed bean correspondiente al Ejemplo 02.

```
package jsf.beans;

import java.util.Date;
. . .

@ManagedBean(name="beanDatos")
@RequestScoped
public class BeanDatos{

    private Date fechaIngreso;
    private Date otraFecha;

    // getters and setters

    public void doProcesar(){
        otraFecha = fechaIngreso;
     }

}
```

VALIDACIÓN

- JSF permite validar el contenido de diferentes datos.
- Podemos usar el atributo validator para apuntar al método de una bean que realiza validación.
- Los validadores se pueden registrar en las clases que implementan EditableValueHolder.
- En caso de no cumplirse la validación han de lanzar ValidatorException conteniendo un FacesMassage con el error.

VALIDACIÓN

Validar Longitud

```
<h:outputLabel value="Nombre:"/>
<h:inputText
   id="nombre" label="Nombre" value="#{empleado.nombre}"
   required="true" requiredMessage="Nombre es obligatorio.">
   <f:validateLength for="nombre" minimum="5" maximum="30"/>
</h:inputText>
<h:message for="nombre" errorStyle="color:red"/>
```

DEMO

JAVA SERVER FACES

Gracias

Eric Gustavo Coronel Castillo
gcoronelc.blogspot.com
gcoronelc@gmail.com