Groovy & Grails: Desarrollo rápido de aplicaciones

Sesión 14: AJAX

AJAX

- Frameworks AJAX
- Ejemplos sencillos
- Ejemplos de uso

Frameworks AJAX

- Grails dispone de varios frameworks AJAX por defecto:
  - Prototype
  - YUI (Yahoo User Interface)
  - Script.aculo.us
  - Dojo
  - Google Web Toolkit

Frameworks AJAX

- Para utilizar cualquiera de estos frameworks, al inicio de nuestras páginas se debe poner la etiqueta

```
<g:javascript>
```

- Se debe especificar el parámetro *library* con el nombre de la librería AJAX en cuestión

```
<g:javascript library="prototype">
<g:javascript library="scriptaculous">
<g:javascript library="yui">
<g:javascript library="dojo">
```

Frameworks AJAX

- *Prototype* y *Scriptaculous* vienen instalados por defecto en Grails

- Para instalar los otros frameworks podemos utilizar la línea de comandos

```
grails install-plugin dojo

grails install-plugin yui
```

Frameworks AJAX

- Cada framework tiene sus propios componentes y su propia forma de trabajar
- Grails nos ofrece una capa superior para trabajar de la misma forma con todos los frameworks

Frameworks AJAX

- Tenemos los siguientes métodos comunes

Método	Descripción
remoteField	Crea un campo de texto que envía su valor a un enlace remoto cuando éste cambia su valor
remoteFunction	Crea una llamada a un método remoto en javascript que puede ser asignada a un evento del DOM
remoteLink	Crea un enlace que llama a una función remota
formRemote	Crea un formulario que ejecuta una llamada AJAX cuando se envía el formulario
javascript	Carga una determinada función Javascript
submitToRemote	Crea un botón que envía una llamada como una función remota

Ejemplos sencillos

- **Etiqueta remoteField**
  - Crea un elemento de formulario de tipo texto que permite invocar un enlace cuando éste cambia su valor
  - Admite los siguientes parámetros

Ejemplos sencillos

- **Etiqueta remoteField**

Parámetro	Obligatorio	Descripción
name	Sí	Especifica el nombre del elemento del formulario
value	No	El valor inicial para el elemento del formulario
paramName	No	El nombre del parámetro enviado al servidor
action	No	El nombre de la acción a utilizar con el enlace. En caso de que no se especifique nada, se utilizará la acción por defecto

Ejemplos sencillos

- **Etiqueta remoteField**

Parámetro	Obligatorio	Descripción
controller	No	El nombre del controlador a utilizar con el enlace. En caso de que no se especifique nada, se utilizará el controlador actual
id	No	El *id* utilizado en el enlace
update	No	Contendrá, bien un mapa con los elementos a actualizar en caso de éxito o fallo en la operación, o una cadena con el elemento a actualizar.
before	No	Una función javascript que se invocará antes de realizar la llamada a la función remota

Ejemplos sencillos

- **Etiqueta remoteField**

Parámetro	Obligatorio	Descripción
after	No	Una función javascript que se invocará después de realizar la llamada a la función remota
asynchronous	No	Indica si la llamada se realiza de forma asíncrona o no. Por defecto este valor es true
method	No	El método a utilizar al realizar la llamada. Por defecto se utilizar el método POST

Ejemplos sencillos

- **Etiqueta remoteField**
  - Vamos a comprobar la disponibilidad de los nombres de usuario cuando se registran nuevos usuarios
  - Necesitamos editar la propiedad *login* de la vista *register.gsp*

Ejemplos sencillos

- **Etiqueta remoteField**

```
<tr class="prop">
        <td valign="top" class="name">
                <label for="login">Login:</label>
        </td>
        <td valign="top" class="value $
{hasErrors(bean:usuarioInstance,field:'login','errors')}">
                <g:remoteField action="checkLogin"
update="spanCheckLogin" name="login" paramName="login"/><span
id="spanCheckLogin"></span>
        </td>
</tr>
```

Ejemplos sencillos

- **Etiqueta remoteField**
  - En la etiqueta *remoteField* hemos definido los parámetros *action* que contiene el nombre del método del controlador de la clase Usuario que se encargará de la comprobación
  - En el parámetro *update* indicamos donde vamos a escribir el resultado de la comprobación
  - Indicamos también el nombre del elemento del formulario y el nombre del parámetro enviado al servidor con *paramName*

Ejemplos sencillos

- **Etiqueta remoteField**

```
def checkLogin = {
        def usuario = Usuario.findByLogin(params.login)
        if (!usuario)
                render("OK")
        else
                render("Nombre de usuario escogido por otro usuario")
}
```

Ejemplos sencillos

- **Etiqueta remoteFunction**
  - Permite especificar una función remota que se invocará cuando se produzca un evento del DOM
  - Admite los siguientes parámetros

Ejemplos sencillos

- **Etiqueta remoteFunction**

Parámetro	Obligatorio	Descripción
action	No	El nombre de la acción a utilizar con el enlace. En caso de que no se especifique nada, se utilizará la acción por defecto
controller	No	El nombre del controlador a utilizar con el enlace. En caso de que no se especifique nada, se utilizará el controlador actual
id	No	El *id* utilizado en el enlace
update	No	Contendrá, bien un mapa con los elementos a actualizar en caso de éxito o fallo en la operación, o una cadena con el elemento a actualizar.

Ejemplos sencillos

- **Etiqueta remoteFunction**

Parámetro	Obligatorio	Descripción
before	No	Una función javascript que se invocará antes de realizar la llamada a la función remota
after	No	Una función javascript que se invocará después de realizar la llamada a la función remota
asynchronous	No	Indica si la llamada se realiza de forma asíncrona o no. Por defecto este valor es *true*
method	No	Indica si la llamada se realiza de forma asíncrona o no. Por defecto este valor es true
params	No	Parámetros para enviar al controlador

Ejemplos sencillos

- **Etiqueta remoteFunction**
  - Modificamos el ejemplo anterior para que ahora se utilice la etiqueta *remoteFunction*

```
<tr class="prop">
        <td valign="top" class="name"> <label for="login">Login:</label> </td>
        <td valign="top" class="value ${hasErrors(bean:usuarioInstance,field:'login','errors')}">
                <input type="text" maxlength="20" id="login" name="login" value="" onChange="${remoteFunction(action:"checkLogin", update:"spanCheckLogin", params:"'login="+this.value+"'")}"/><span id="spanCheckLogin"></span>
        </td>
</tr>
```

Ejemplos sencillos

- **Etiqueta remoteFunction**
  - El método *checkLogin()* quedaría como antes

Ejemplos sencillos

- **Etiqueta remoteLink**
  - Permite realizar llamadas a una función remota para realizar una determinada acción a partir de un enlace
  - Se suele utilizar para la recarga parcial de determinadas partes de una página
  - Tenemos los siguientes parámetros

Ejemplos sencillos

- **Etiqueta remoteLink**

Parámetro	Obligatorio	Descripción
action	No	El nombre de la acción a utilizar con el enlace. En caso de que no se especifique nada, se utilizará la acción por defecto
controller	No	El nombre del controlador a utilizar con el enlace. En caso de que no se especifique nada, se utilizará el controlador actual
id	No	El *id* utilizado en el enlace
params	No	Parámetros para enviar al controlador en forma de mapa

Ejemplos sencillos

- **Etiqueta remoteLink**

Parámetro	Obligatorio	Descripción
update	No	Contendrá, bien un mapa con los elementos a actualizar en caso de éxito o fallo en la operación, o una cadena con el elemento a actualizar.
before	No	Una función javascript que se invocará antes de realizar la llamada a la función remota
after	No	Una función javascript que se invocará después de realizar la llamada a la función remota

Ejemplos sencillos

- **Etiqueta remoteLink**

Parámetro	Obligatorio	Descripción
asynchronous	No	Indica si la llamada se realiza de forma asíncrona o no. Por defecto este valor es true
method	No	El método a utilizar al realizar la llamada. Por defecto se utilizar el método POST

Ejemplos sencillos

- **Etiqueta remoteLink**
  - Imagina una serie de pestañas y al hacer clic sobre ellas se muestra su información

```
<ul>
        <li><g:remoteLink action="showTab" params="[id:'1']"
update="tabcontent">Primero</g:remoteLink></li>
        <li><g:remoteLink action="showTab" params="[id:'2']"
update="tabcontent">Segundo</g:remoteLink></li>
        <li><g:remoteLink action="showTab" params="[id:'3']"
update="tabcontent">Tercero</g:remoteLink></li>
</ul>
<div id="tabcontent"></div>
```

Ejemplos sencillos

- **Etiqueta remoteLink**
  - Añadimos el método *showTab()* al controlador correspondiente

```
def showTab = {
        render("Contenido ${params.id}")
}
```

Ejemplos sencillos

- **Etiqueta formRemote**
  - Creamos un formulario que se ejecutará remotamente al enviarlo
  - Admite los siguientes parámetros

Ejemplos sencillos

- **Etiqueta formRemote**

Parámetro	Obligatorio	Descripción
url	Sí	La URL que se encargará de gestionar el formulario. Se especifica en forma de mapa de valores con la acción, controlador e identificador
name	No	El nombre del formulario
action	No	El nombre de la acción que se ejecutará cuando se vuelva de ejecutar el formulario remoto
update	No	Contendrá, bien un mapa con los elementos a actualizar en caso de éxito o fallo en la operación, o una cadena con el elemento a actualizar.

Ejemplos sencillos

- **Etiqueta formRemote**

Parámetro	Obligatorio	Descripción
before	No	Una función javascript que se invocará antes de realizar la llamada a la función remota
after	No	Una función javascript que se invocará después de realizar la llamada a la función remota
asynchronous	No	Indica si la llamada se realiza de forma asíncrona o no. Por defecto este valor es true
method	No	El método a utilizar al realizar la llamada. Por defecto se utilizar el método POST

Ejemplos sencillos

- **Etiqueta formRemote**
  - Creamos un ejemplo para identificar a los usuarios de nuestra aplicación de forma remota
  - Modificamos el archivo *login.gsp*

```
<g:formRemote name="miForm" update="content" action="list" url="$
{[action:'handleLogin']}">
        Login: <input name="login" type="text"/>
        Password: <input name="password" type="password"/>
        <input type="submit" value="Enviar"/>
</g:formRemote>
<div id="content"></div>
```

Ejemplos sencillos

- **Etiqueta javascript**
  - Podemos incluir funciones javascript de tres formas diferentes gracias a los parámetros

Parámetro	Obligatorio	Descripción
library	No	El nombre de la librería a incluir. Puede ser *prototype*, *scriptaculous*, *yui* o *dojo*
src	No	El nombre del archivo javascript a incluir. Se buscará este archivo en el directorio */app/js*
base	No	Permite indicarle una ruta absoluta para cargar el archivo js correspondiente

Ejemplos sencillos

- **Etiqueta submitToRemote**
  - Tiene la misma funcionalidad que *formRemote,* pero creando un botón de tipo *submit* que enviará los datos introducidos a una función remota donde serán analizados
  - Admite los siguientes parámetros

Ejemplos sencillos

- **Etiqueta submitToRemote**

Parámetro	Obligatorio	Descripción
url	Sí	La URL que se encargará de gestionar el formulario. Se especifica en forma de mapa de valores con la acción, controlador e identificador
action	No	El nombre de la acción que se ejecutará cuando se vuelva de ejecutar el formulario remoto
update	No	Contendrá, bien un mapa con los elementos a actualizar en caso de éxito o fallo en la operación, o una cadena con el elemento a actualizar.
before	No	Una función javascript que se invocará antes de realizar la llamada a la función remota

Ejemplos sencillos

- **Etiqueta submitToRemote**

Parámetro	Obligatorio	Descripción
after	No	Una función javascript que se invocará después de realizar la llamada a la función remota
asynchronous	No	Indica si la llamada se realiza de forma asíncrona o no. Por defecto este valor es true
method	No	Indica si la llamada se realiza de forma asíncrona o no. Por defecto este valor es true

Ejemplos sencillos

- **Etiqueta submitToRemote**
  - Modificamos la página *login.gsp* para sustituir el botón *submit* por otro que actúe de forma remota

```
<div class="buttons">
        <span class="button">
                <g:submitToRemote update="content" url="$
{[action:'handleLogin']}" value="Enviar"/>
        </span>
        <div id="content"></div>
</div>
```

Ejemplos de uso

- editInPlace
- Autocompletados
- Sistema de votos con estrellas

Ejemplos de uso

- **editInPlace**
  - Habitualmente, cuando queremos modificar de una determinada propiedad, debemos modificar el objeto completo
  - Para evitar esto, vamos a darle al usuario la posibilidad de editar sólo la propiedad que desea modificar

Ejemplos de uso

- **editInPlace**
  - Vamos a hacer esto con la propiedad *título* de los libros
  - Necesitamos crear una nueva librería de etiquetas que llamaremos *AjaxTagLib*

Ejemplos de uso

- **editInPlace**

```
class AjaxTagLib {
        def editInPlace = {attrs, body ->
                def rows = attrs.rows ? attrs.rows : 0;
                def cols = attrs.cols ? attrs.cols : 0;
                def id = attrs.remove('id')
                out << "<span id='${id}'>"
                out << body()
                out << "</span>"
                out << "<script type='text/javascript'>"
                out << "new Ajax.InPlaceEditor('${id}', '"
    ....
}
```

Ejemplos de uso

- **editInPlace**

```
class AjaxTagLib {
        def editInPlace = {attrs, body ->
                ....
                out << createLink(attrs)
                out << "',{" if(rows)
                out << "rows:${rows},"
                if(cols)
                        out << "cols:${cols},"
                ....
        }
}
```

Ejemplos de uso

- **editInPlace**

```
class AjaxTagLib {
        def editInPlace = {attrs, body ->
                ....
                if(attrs.paramName) {
                        out << """callback: function(form, value) {
                        return '${attrs.paramName}=' + escape(value) }"""
                }
                out << "});"
                out << "</script>"
        }
}
```

Ejemplos de uso

- **editInPlace**
  - Se utiliza una función de *scriptaculous* llamada *Ajax.InPlaceEditor*
  - Debemos añadir la nueva etiqueta en el listado de libros

```
<g:editInPlace id="listlibros${libroInstance.id}" url="[controller:'libro',
action:'editTitulo', id:libroInstance.id]" rows="1" cols="10" paramName="titulo">
        ${libroInstance.titulo}
</g:editInPlace>
```

Ejemplos de uso

- **editInPlace**
  - Debemos crear también el método *editTitulo()* en el controlador de la clase *Libro*

```
def editTitulo = {
        def libro = Libro.get(params.id)
        libro.titulo = params.titulo

        libro.save()

        render params.titulo
}
```

Ejemplos de uso

- **editInPlace**
  - El método *editTitulo()* recoge la información del libro a partir del *id*
  - Posteriormente modifica su valor
  - Por último, devuelve el valor escrito para que se muestre la modificación

Ejemplos de uso

- **Autocompletados**
  - Los autocompletados es algo que cada vez abunda más en las aplicaciones web
  - Google lo ha incorporado recientemente en sus búsquedas
  - Supone una ayuda extra al usuario

Ejemplos de uso

- **Autocompletados**
  - En nuestra aplicación, un buen lugar para añadir autocompletados sería utilizarlo en la propiedad *autor* al editar o crear un *libro*
  - De esta forma, el usuario tiene una ayuda para no tener que escribir el nombre completo del autor en caso de que ya esté insertado en la base de datos

Ejemplos de uso

- **Autocompletados**
  - Vamos a utilizar de nuevo el plugin *RichUI* para implementar esta característica en nuestro sistema
  - Para utilizar el autocompletado debemos incluir la etiqueta *<resource:autoComplete skin="default"/>*
  - Debemos sustituir la caja de texto correspondiente por la siguiente etiqueta

```
<richui:autoComplete name="autor" action="${createLinkTo('dir': 'libro/
showAutores')}" />
```

Ejemplos de uso

- **Autocompletados**
  - Definimos el nombre del elemento del formulario (*autor*) y la acción que se encarga de realizar la búsqueda de autores (*libro/showAutores*)
  - Nos queda definir el método *showAutores()* en el controlador de la clase *Libro*

Ejemplos de uso

- **Autocompletados**

```
def showAutores = {
        def libros = Libro.createCriteria().listDistinct {
                ilike("autor","%${params.query}%")
                order("autor")
                maxResults(5)
        }
        render(contentType: "text/xml") {
                results() {
                        libros.each { libro ->
                                result(){ name(libro.autor) }
                        }
                }
        }
}
```

Ejemplos de uso

- **Autocompletados**

```
<results>
        <result>
                <name>
                        Miguel de Cervantes Saavedra
                </name>
        </result>
        <result>
                <name>
                        Camilo José Cela Trúlock
                </name>
        </result>
</results>
```

Ejemplos de uso

- **Autocompletados**

Ejemplos de uso

- **Sistema de votos con estrellas**
  - Consiste en permitir al usuario la valoración de los contenidos de una aplicación
  - La página oficial de plugins de Grails tiene un sistema de este tipo
  - El plugin *RichUI* nos va a permitir hacer esto gracias a su componente *Star rating*

Ejemplos de uso

- **Sistema de votos con estrellas**
  - Vamos a permitir a los usuarios de nuestra aplicación que valoren los libros de la biblioteca
  - Debemos incluir la siguiente etiqueta en la cabecera de las páginas donde vayamos a implementar este sistema

```
<resource:rating/>
```

Ejemplos de uso

- **Sistema de votos con estrellas**
  - Debemos añadir a la clase *Libro* un par de propiedades para controlar la media de votos y el número total de votos recibidos

```
class Libro {
        String isbn
        String titulo
        ….
        Double valoracion = 0
        Integer totalVotos = 0
        .....
}
```

Ejemplos de uso

- **Sistema de votos con estrellas**
  - Debemos crear una plantilla donde mostrar las estrellas y que los usuarios puedan votar y consultar las votaciones ya realizadas (*grails-app/views/libro/_rate.gsp)*

Ejemplos de uso

- **Sistema de votos con estrellas**

```
<div id="libro${libroInstance.id}">
        <richui:rating dynamic="true" id="${libroInstance.id}" units="5"
rating="${valoracion}" updateId="libro${libroInstance.id}" controller="libro"
action="votar" />
        <p class="static"> Valoración $
{java.text.NumberFormat.instance.format(libroInstance.valoracion)}
        sobre un total de ${libroInstance.totalVotos} voto
        <g:if test="${libroInstance.totalVotos != 1}">s</g:if>
        </p>
</div>
```

Ejemplos de uso

- **Sistema de votos con estrellas**
  - La etiqueta <*richui:rating/*> tiene los siguientes parámetros
    - *dynamic,* indica si el usuario puede votar
    - *id*, almacena el identificador del libro
    - *units*, indica el número de estrellas a mostrar
    - *rating,* valoración media del libro
    - *updateId,* indica el elemento del DOM a actualizar
    - *controller,* el controlador que se encarga de la gestión del voto
    - *action,* la acción que se encarga de la gestión del voto

Ejemplos de uso

- **Sistema de votos con estrellas**
  - El usuario verá el sistema de estrellas cuando vea la información del libro
  - Debemos modificar el archivo *grails-app/views/libro/show.gsp*

Ejemplos de uso

- **Sistema de votos con estrellas**

```
<html>
        <head>
            <meta http-equiv="Content-Type" content="text/html; charset=UTF-8"/>
            <meta name="layout" content="main" />
            <resource:rating />
            <title>Show Libro</title>
        </head>
        .......
        <tr class="prop"> <td valign="top" class="name" colspan="2">
            <g:render template="rate" model='[libro: libroInstance,
valoracion: "${libroInstance.valoracion}"]' />
        </td> </tr>
</html>
```

Ejemplos de uso

- **Sistema de votos con estrellas**
  - Ya sólo nos queda implementar el método *votar()* del controlador de la clase *Libro*
  - Este método recibe el identificador del libro y la valoración del usuario actual

Ejemplos de uso

- **Sistema de votos con estrellas**

```
def votar = {
    def valoracion = params.rating
    def libro = Libro.get( params.id )
    def media = (valoracion.toDouble() + libro.valoracion*libro.totalVotos)/
(libro.totalVotos + 1)
    libro.valoracion = media
    libro.totalVotos += 1
    libro.save()
    render(template: "/libro/rate", model: [libroInstance: libro, valoracion: media])
}
```

Ejemplos de uso

- **Sistema de votos con estrellas**