

¿Qué ofrece Autentia Real Business Solutions S.L?

Somos su empresa de **Soporte a Desarrollo Informático**.
 Ese apoyo que siempre quiso tener...

1. Desarrollo de componentes y proyectos a medida



2. Auditoría de código y recomendaciones de mejora

3. Arranque de proyectos basados en nuevas tecnologías

1. Definición de frameworks corporativos.
2. Transferencia de conocimiento de nuevas arquitecturas.
3. Soporte al arranque de proyectos.
4. Auditoría preventiva periódica de calidad.
5. Revisión previa a la certificación de proyectos.
6. Extensión de capacidad de equipos de calidad.
7. Identificación de problemas en producción.



4. Cursos de formación (impartidos por desarrolladores en activo)

Spring MVC, JSF-PrimeFaces /RichFaces,
 HTML5, CSS3, JavaScript-jQuery

Gestor portales (Liferay)
 Gestor de contenidos (Alfresco)
 Aplicaciones híbridas

Tareas programadas (Quartz)
 Gestor documental (Alfresco)
 Inversión de control (Spring)

Control de autenticación y
 acceso (Spring Security)
 UDDI
 Web Services
 Rest Services
 Social SSO
 SSO (Cas)

JPA-Hibernate, MyBatis
 Motor de búsqueda empresarial (Solr)
 ETL (Talend)

Dirección de Proyectos Informáticos.
 Metodologías ágiles
 Patrones de diseño
 TDD

BPM (jBPM o Bonita)
 Generación de informes (JasperReport)
 ESB (Open ESB)

Estás en: Inicio Tutoriales Obtención de la fila seleccionada en un datatable con JSF2

	<p>DESARROLLADO POR:  Jose Manuel Sánchez Suárez </p>
<p>Consultor tecnológico de desarrollo de proyectos informáticos.</p> <p>Puedes encontrarme en Autentia: Ofrecemos servicios de soporte a desarrollo, factoría y formación</p> <p>Somos expertos en Java/J2EE</p>	

Catálogo de servicios Autentia

[Anuncios Google](#) [Tutorial](#) [C# PDF Tutorial](#) [PHP Form Tutorial](#)

Fecha de publicación del tutorial: 2011-02-08



Share |

[Regístrate para votar](#)

Obtención de la fila seleccionada en un datatable con JSF2.

0. Índice de contenidos.

- 1. Introducción.
- 2. Entorno.
- 3. Invocación al método getRowData.
- 4. Enviando el identificador del objeto como parámetro.
- 5. Enviando el objeto como parámetro.
- 6. Con el soporte de Jboss Seam.
- 7. Referencias.
- 8. Conclusiones.

1. Introducción

En este tutorial vamos a analizar las opciones que tenemos disponibles en JSF2 para obtener el registro de la fila seleccionada en un dataTable.

La historia de usuario que queremos cumplir sería similar a esta: **Como Pablo quiero acceder, desde el listado de usuarios, a la edición de los mismos para modificar sus características de acceso.** Pablo es el administrador de usuarios en la aplicación de nuestro hipotético cliente.

Vamos a examinar todas las posibilidades de las que disponemos, que no son pocas, haciendo uso del estándar, y con el soporte de anotaciones de Jboss Seam. Existen más posibilidades si hacemos uso de librerías de componentes visuales de terceros como ICEfaces, RichFaces o Primefaces, pero dichas posibilidades las vamos a examinar en otro tutorial en el que hablaremos de la posibilidad de selección múltiple de filas de un dataTable.

Para el objetivo del tutorial, la selección de un registro de la fila puede suponer navegación a otra vista o no, en función de si el evento de selección responde a un listener o a un action.

2. Entorno.

El tutorial está escrito usando el siguiente entorno:

- Hardware: Portátil MacBook Pro 17' (2.93 GHz Intel Core 2 Duo, 4GB DDR3 SDRAM).

[UML 2.0 Tutorial](#)
Últimas Noticias

[XIII Charla Autentia - AOS y TDD - Vídeos y Material](#)

[Las metodologías ágiles como el catalizador del cambio](#)

[XIV Charla Autentia - ZK](#)

[Informática profesional: Las reglas no escritas para triunfar en la empresa. 2ª EDICIÓN ACTUALIZADA.](#)

[Pequeño coding dojo con Carlos Ble en las oficinas de Autentia.](#)

[Histórico de NOTICIAS](#)

Últimos Tutoriales

[Eclipse custom templates](#)

[Haaala! para Android: Facebook y Email con Voz](#)

[Librería de acceso a datos con Spring y JPA](#)

- Sistema Operativo: Mac OS X Snow Leopard 10.6.1
- JSF2 (Mojarra 2.0.4)
- Jboss Seam 2.1 y RichFaces 3.3
- Apache Tomcat 7.0.6

 Como editar XML o HTML con el plugin xmledit de Vim

 Notificación de eventos en Nut

3. Invocación al método `getRowData`.

La primera opción que vamos a examinar es la posibilidad de parametrizar a nivel de `dataTable` y mediante el atributo `binding`, la referencia a un objeto del lado del `managedBean`. En dicho objeto, representado por la clase `UIData`, se mantiene una referencia del `dataTable` que se almacena en el árbol de componentes, que vive en el `FacesContext` durante toda la request.

De este modo, en la vista tendremos algo similar a lo siguiente:

```

01 <h:dataTable id="usersList"
02     var="user"
03     value="#{usersView.users}"
04     rows="10"
05     binding="#{usersView.usersDataTable}">
06 <h:column>
07     <f:facet name="header">
08         <h:outputText id="number_label"
09             value="#{msg['Users.login']}'"/>
10     </f:facet>
11     <h:commandLink actionListener="#{usersView.editUser}" immediate="true">
12         <f:ajax render="@form"/>
13         <h:outputText id="login"
14             value="#{user.login}"/>
15     </h:commandLink>
16 </h:column>
17     ...
18 </h:dataTable>

```

Y en el `managedBean` el método que responde al evento obtiene el registro seleccionado de la siguiente forma:

```

01 @ManagedBean
02 @ViewScoped
03 public class UsersView {
04     ...
05     private UIData usersDataTable;
06
07     public void setUsersDataTable(UIData usersDataTable) {
08         this.usersDataTable = usersDataTable;
09     }
10
11     public UIData getUsersDataTable() {
12         return usersDataTable;
13     }
14     ...
15     public void editUser(ActionEvent event){
16         log.trace("User selected " + ((User) usersDataTable.getRowData()) );
17     }
18     ...
19 }
20

```

Sería indiferente si trabajásemos con un `action` o con un `actionListener`, funciona con los dos tipos de evento.

4. Enviando el identificador del objeto como parámetro.

También podemos enviar como parámetro el identificador del usuario, de modo que se añada al mapa de parámetros de la request y obtenerlo desde el método correspondiente del `managedBean`.

Para ello desde la vista usamos el componente no visual `<f:param>`

```

01 <h:dataTable id="usersList"
02     var="user"
03     value="#{usersView.users}"
04     rows="10"
05     binding="#{usersView.usersDataTable}">
06 <h:column>
07     <f:facet name="header">
08         <h:outputText id="number_label"
09             value="#{msg['Users.login']}'"/>
10     </f:facet>
11     <h:commandLink actionListener="#{usersView.editUser}" immediate="true">
12         <f:param name="userLogin" value="#{user.login}" />
13         <f:ajax render="@form"/>
14         <h:outputText id="login"
15             value="#{user.login}"/>
16     </h:commandLink>
17 </h:column>
18     ...
19 </h:dataTable>

```

En el método que recibe el evento podemos obtener el parámetro como sigue:

```

01 @ManagedBean
02 @ViewScoped

```

Últimos Tutoriales del Autor

-  Eclipse custom templates
-  Zen-coding: una nueva forma de escribir código HTML
-  IAQ (Interesting Asked Questions), implementado una interfaz SPI con jQuery
-  Google Chrome Developer Toolbar.
-  Introducción a HTML5.

Síguenos a través de:



Últimas ofertas de empleo

- 2010-10-11  Comercial - Ventas - SEVILLA.
- 2010-08-30  Otras - Electricidad - BARCELONA.
- 2010-08-24  Otras Sin catalogar - LUGO.
- 2010-06-25  T. Información - Analista / Programador - BARCELONA.

 Jose Manuel Sánchez
sanchezsuares

```

03 public class UsersView {
04     ...
05     private UIData usersDataTable;
06     ...
07     public void editUser(ActionEvent event){
08         final String userLogin =
FacesContext.getCurrentInstance().getExternalContext().getRequestParameterMap().get("userLogin");
09         selectedUser = FinancialService.getInstance().getByLogin(userLogin);
10         log.trace("User selected " + selectedUser );
11     }
12     ...
13 }
14

```

Aquí el inconveniente es que tenemos que volver a obtener el objeto totalmente poblado porque lo que remitimos es su identificador y se trata de un tipo básico.

5. Enviando el objeto como parámetro.

Existe una alternativa a la de remitir el identificador que es la de remitir todo el objeto completo y, para ello tenemos dos posibilidades.

La primera es hacer uso del componente no visual <f:setPropertyActionListener en el que le podemos indicar el valor que tendrá un atributo del managedBean antes de que el evento de acción sea invocado.

```

01 <h:dataTable id="usersList"
02     var="user"
03     value="#{usersView.users}"
04     rows="10"
05     binding="#{usersView.usersDataTable}">
06 <h:column>
07     <f:facet name="header">
08         <h:outputText id="number_label"
09             value="#{msg['Users.login'] }"/>
10     </f:facet>
11     <h:commandLink actionListener="#{usersView.editUser}" immediate="true">
12         <f:setPropertyActionListener value="#{user}"
13             target="#{usersView.selectedUser}" />
14     <f:ajax render="@form"/>
15     <h:outputText id="login"
16         value="#{user.login}"/>
17     </h:commandLink>
18 </h:column>
19     ...
20 </h:dataTable>

```

Del lado del managedBean solo necesitaremos el atributo con sus métodos set & get.

```

01 @ManagedBean
02 @ViewScoped
03 public class UsersView {
04     ...
05     private User selectedUser;
06
07     public void setSelectedUser(User selectedUser) {
08         this.selectedUser = selectedUser;
09     }
10
11     public User getSelectedUser() {
12         return selectedUser;
13     }
14     ...
15 }
16
17

```

La segunda opción, solo podremos hacer uso de ella, si tenemos disponible la versión 2.2 de EL, como en nuestro caso que usamos Tomcat 7.

Consiste en invocar directamente desde Expression Language a un método del managedBean pasándole un parámetro.

```

01 <h:dataTable id="usersList"
02     var="user"
03     value="#{usersView.users}"
04     rows="10"
05     binding="#{usersView.usersDataTable}">
06 <h:column>
07     <f:facet name="header">
08         <h:outputText id="number_label"
09             value="#{msg['Users.login'] }"/>
10     </f:facet>
11     <h:commandLink action="#{usersView.editUser(user)}" immediate="true">
12         <f:ajax render="@form"/>
13         <h:outputText id="login"
14             value="#{user.login}"/>
15     </h:commandLink>
16 </h:column>
17     ...
18 </h:dataTable>

```

De este modo, en el managedBean, tendríamos simplemente el método con dicho parámetro de entrada.

@franciscoferri ya, pero lo acaparan mi mujer y mi niño.

54 minutes ago · reply

Quiero el iPad que los de @atSistemas sortean en la conferencia Spring I/O
<http://bit.ly/g7jITP>
#springio #java #groovy

2 hours ago · reply

Eclipse custom templates:
<http://bit.ly/fjnwLU>
8 hours ago · reply

Tenemos más capacidad de olvidar que de recordar. Si

 Join the conversation

```

01 @ManagedBean
02 @ViewScoped
03 public class UsersView {
04     ...
05     public void editUser(User user){
06         selectedUser = user;
07         log.trace("User selected " + selectedUser );
08     }
09     ...
10 }
11

```

Esta técnica es la que veníamos usando hasta ahora en los proyectos que tenían el soporte de Jboss Seam que sí permitía invocar a un método del managedBean pasándole parámetros de entrada.

6. Con el soporte de Jboss Seam.

Además de la que hemos comentado anteriormente existe una segunda posibilidad en Jboss Seam consistente en mantener en el managedBean un atributo anotado que tendrá una referencia al dataTable y un segundo atributo, también anotado, que mantiene una referencia a la fila seleccionada.

```

01 @Name("usersView")
02 @Scope(ScopeType.SESSION)
03 public class UsersView {
04     ...
05     @DataModel
06     List<User> users;
07     ...
08     @DataModelSelection
09     User selectedUser;
10     ...
11 }
12

```

La vista no requiere de ninguna configuración adicional, desde cualquier método de evento del managedBean podemos acceder al atributo que estará poblado con la entidad de la fila seleccionada.

```

1  ...
2  <rich:dataTable id="usersList"
3      var="user"
4      value="#{usersView.users}"
5      rows="10">
6  ...

```

7. Referencias.

- <http://balusc.blogspot.com/2006/06/using-datatables.html>

8. Conclusiones.

Hemos visto que no son pocas las posibilidades que tenemos en nuestra mano, hacer o haber hecho uso de una u otra dependerá de nuestro grado de acercamiento al api, esto es, del grado de madurez que tengamos con JSF.

Un saludo.

Jose

jmsanchez@autentia.com

Anímate y coméntanos lo que pienses sobre este **TUTORIAL**:

Puedes opinar o comentar cualquier sugerencia que quieras comunicarnos sobre este tutorial; con tu ayuda, podemos ofrecerte un mejor servicio.

Enviar comentario

(Sólo para usuarios registrados)

» **Regístrate** y accede a esta y otras ventajas «

COMENTARIOS



Esta obra está licenciada bajo licencia Creative Commons de Reconocimiento-No comercial-Sin obras derivadas 2.5

Copyright 2003-2011 © All Rights Reserved | [Textos](#) | [Condiciones de uso](#) | [Banners](#) | [Powered by Autentia](#) |

