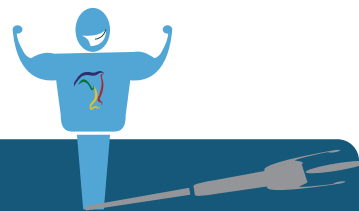


¿Qué ofrece Autentia Real Business Solutions S.L?

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

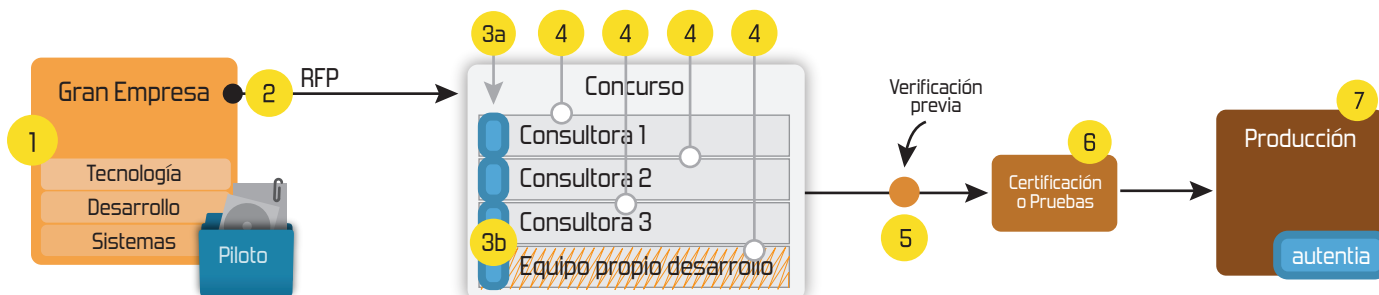
1. Desarrollo de componentes y proyectos a medida



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)

Estas en: **Inicio** **Tutoriales** Facelets en JSF 2: sistema de plantillas y componentes por composición.

Últimas Noticias

- » [VII Charla Autentia - Pluto](#)
- » [Competición Plasma Cars \(Autos Locos\) - SEGUNDO INTENTO](#)
- » [Probando con Marick - Fotos y vídeo](#)
- » [Competición Plasma Cars \(Autos Locos\) - EVENTO POSPUESTO](#)
- » [VI Charla Autentia: Mapeos en Hibernate - Vídeos y Material](#)
- » [Competición Plasma Cars \(Autos Locos\) - EVENTO POSPUESTO](#)

+Noticias Destacadas

- » [VII Charla Autentia - Pluto](#)
- » [Competición Plasma Cars \(Autos Locos\) - SEGUNDO INTENTO](#)
- » [Probando con Marick - Fotos y vídeo](#)
- » [Autentia estuvo en el Spring 2GX Day](#)

+Comentarios Cómic

+Enlaces

Catálogo de servicios Autentia



Triptico
(6,3 MB)



Cómic
(3,1 MB)

Tutorial desarrollado por



Jose Manuel Sánchez Suárez

Consultor tecnológico de desarrollo de proyectos informáticos.

Puedes encontrarme en [Autentia](#)

Somos expertos en Java/J2EE

Catálogo de servicios de Autentia

[Descargar \(6,3 MB\)](#)

[Descargar en versión comic \(3,1 MB\)](#)

[AdictosAlTrabajo.com](#) es el Web de difusión de conocimiento de [Autentia](#).



[Catálogo de cursos](#)

Descargar este documento en formato PDF: [jsf2FaceletsTemplatesAndCompositeComponents.pdf](#)

Acceso de usuarios registrados:

E-mail:

Contraseña:

[Entrar](#)

[Deseo registrarme](#)

[He olvidado mis datos de acceso](#)

Registra tu empresa:

Descubre las ventajas de registrar tu empresa en [AdictosAlTrabajo...](#)

[Registrar mi empresa](#)

[Listado de empresas ya registradas](#)



☐ Web

☒ [www.adictosaltrabajo.com](#)

[Buscar](#)

Últimos tutoriales

2010-04-20
[Facelets en JSF 2: sistema de plantillas y componentes por composición.](#)

2010-04-19
[DbVisualizer free version.](#)

2010-04-09
[Session TimeOut en RichFaces, con el soporte de Jboss Seam.](#)

2010-04-08
[Jetspeed-2 de Apache Software Foundation](#)

Fecha de creación del tutorial: 2010-04-20

Facelets en JSF 2: sistema de plantillas y componentes por composición.

0. índice de contenidos.

- [1. Introducción.](#)
- [2. Entorno.](#)
- [3. El sistema de plantillas de Facelets.](#)
 - [3.1. Definición de la plantilla de la aplicación.](#)
 - [3.2. Definición de la plantilla de menú.](#)
 - [3.3. Uso de la plantilla.](#)
- [4. Componentes por composición.](#)
 - [4.1. Configuración.](#)
 - [4.2. Creación de un componente por composición.](#)
 - [4.3. Uso de un componente por composición.](#)
 - [4.4. Exponer las propiedades y comportamientos de los componentes de un componente por composición.](#)
- [5. Referencias.](#)
- [6. Conclusiones.](#)

1. Introducción.

Ahora que ya conocemos [las principales novedades de JSF 2 de la mano de Alejandro](#), en este tutorial vamos a realizar una introducción al uso de Facelets en JSF 2, el sistema de plantillas estándar de facto en la versión 1.2 de JSF y que ahora, en la versión 2, forma parte de la propia especificación.

Con JSF 1.2, la versión anterior, existía la posibilidad de utilizar otros sistemas de plantillas: [Tiles de Apache](#) (siempre más ligado al uso de Struts), [Jsftemplating](#) (asociado a Woodstock) o [JTPL de IBM](#) el sistema de plantillas estándar del entorno de desarrollo de IBM; si bien, para los que apostamos por facelets, al ver las múltiples ventajas frente a estos otros sistemas, ahora nos encontramos con que se soporta por defecto, puesto que ya forma parte del estándar y ya tenemos parte del camino recorrido. Es más:

- ahora Facelets es el sistema por defecto para crear vistas basadas en árboles de componentes JSF, en detrimento de JSP, ya no hay que configurar el soporte de Facelets, al contrario, hay que habilitar el modo de compatibilidad con JSP, si queremos seguir usándolo.
- ninguna de las nuevas características de JSF 2 (el soporte nativo para Ajax, la gestión de eventos del ciclo de vida desde la vista, por supuesto la nueva definición de componentes por composición,...) están disponibles usando JSP.

En resumen, quien no haya [migrado sus vistas de JSP a Facelets](#), ahora se va a perder lo bueno de JSF 2.

Centrándonos en Facelets, a "grosso modo", en JSF 2:

- hace uso del mismo sistema de plantillas que veníamos usando hasta ahora en JSF 1.2, y
- mejora la creación de componentes por composición, reduciendo complejidad en su configuración y añadiendo rigidez en su parametrización, puesto que ahora vamos a tener la posibilidad de definir qué atributos acepta el componente por composición, además de exponer el comportamiento de los componentes que incluye. Esto no significa que los componentes por composición que ya tenemos creados, basándonos en JSF 1.2, los tengamos que re-escribir para JSF 2, porque son totalmente válidos.

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).
- Sistema Operativo: Mac OS X Snow Leopard 10.6.1
- JDK 1.6.0_17.
- Maven 2.2.1.
- Eclipse 3.5: Ganymede, con IAM (plugin para Maven).
- Apache Tomcat 6.0.20.
- JSF 2 (2.0.2)

3. El sistema de plantillas de Facelets.

Como he comentado en la introducción, el sistema de plantillas es el mismo que teníamos con JSF 1.2, y se basa en el uso de las siguientes etiquetas:

- **ui:composition:** envuelve un conjunto de componentes para ser reutilizados en otra página, es la etiqueta que:
 - envuelve o puede envolver la plantilla.
 - se utiliza para hacer referencia a una plantilla.todo lo que quede fuera de la etiqueta ui:composition no será renderizado.
- **ui:define:** define un contenido nombrado para ser insertado en una plantilla, su contenido es un conjunto de componentes y se identifica con un name. Ese conjunto de componentes será insertado en una etiqueta ui:insert con el mismo name.
- **ui:insert:** declara un contenido nombrado que debe ser definido en otra página,
- **ui:decorate:** es la etiqueta que sirve para hacer referencia a una plantilla, como la etiqueta ui:composition, solo que con ui:decorate lo definido antes y después de la etiqueta sí será renderizado,
- **ui:param:** nos sirve para declarar parámetros y su valor en el uso de plantillas y componentes por composición,
- **ui:include:** es una etiqueta que sirve para incluir en una página el contenido de otra, como si el contenido de esta última formase parte de la primera.

3.1. Definición de la plantilla de aplicación.

Siguiendo las recomendaciones de facelets, las plantillas las incluiremos dentro del directorio WEB-INF/facelets/templates/, de modo que no sean visibles desde el contenedor web.

Un ejemplo de plantilla, defaultLayout.xhtml, podría ser el siguiente:

```
01 <?xml version="1.0" encoding="UTF-8"?>
02 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
03 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
04 <html xmlns="http://www.w3.org/1999/xhtml" xmlns:ui="http://java.sun.com/jsf/facelets"
05     xmlns:f="http://java.sun.com/jsf/core" xmlns:h="http://java.sun.com/jsf/html">
06
07 <h:head>
08   <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
09   <h:outputStylesheet library="css" name="style.css" />
10   <title><ui:insert name="title" /></title>
11 </h:head>
12 <h:body>
13   <div id="layout">
14     <div id="header">
15       <div id="logo"></div>
16       <ui:include src="menu.xhtml"/>
17     </div>
18
19     <div id="content">
20       <ui:insert name="content">
21         <span>Página en construcción</span>
22       </ui:insert>
23     </div>
24
25     <div id="footer">
26       <p><a href="#">@</a></p>
27     </div>
28   </div>
29 </h:body>
30 </html>
```

Definimos una maquetación por capas y el contenido de nuestro site en tres secciones: header, content y footer

Prevemos la inclusión de la definición externa de dos contenidos nombrados: title (línea 10) y content (línea 20), de modo que la página que haga uso de la plantilla debe definir tales contenidos. Si no los define podemos prever un valor por defecto, en el caso de content "Página en construcción".

Por último, hemos hecho una inclusión en la línea 16 del contenido de una mini-plantilla, que contendrá el menú de la aplicación.

3.2. Definición de la plantilla de menú.

La plantilla del menú la ubicamos en el mismo directorio que la plantilla de aplicación y podría tener el siguiente contenido:

```
01 <?xml version="1.0" encoding="UTF-8"?>
02 <html xmlns="http://www.w3.org/1999/xhtml" xmlns:ui="http://java.sun.com/jsf/facelets"
03     xmlns:f="http://java.sun.com/jsf/core" xmlns:h="http://java.sun.com/jsf/html"
04     xmlns:p="http://primefaces.prime.com.tr/ui">
05   <ui:composition>
06     <p:menubar>
07       <p:submenu label="#{msg.menu_home}">
08         <p:menuItem label="Home" url="#" />
09       </p:submenu>
10       <p:submenu label="#{msg.menu_items}">
11         <p:menuItem label="#{msg.new}" url="#{menuBean.createItem}"></p:menuItem>
12         <p:menuItem label="#{msg.list}" url="#{menuBean.listItem}"></p:menuItem>
13       </p:submenu>
14     </p:menubar>
15   </ui:composition>
16 </html>
```

Estamos haciendo uso de un componente visual de la [librería de componentes primefaces](#), que soporta JSF 2, que renderiza el contenido de un árbol jerárquico de items como un menú.

2010-04-07
[Primeros pasos con Balsamiq Mockups](#)

2010-03-18
[Revisando los ejemplos de Cocos2d para iPhone.](#)

2010-03-16
[Organización de eventos con StageHQ](#)

2010-03-15
[Retrasar la carga de Javascript con jQuery.getScript\(\).](#)

2010-03-15
[Optimización de páginas web con Page Speed.](#)

2010-03-09
[JSF 2 ya está aquí !!! The JSF Return, ahora más sencillo que nunca !!!](#)

2010-03-08
[Instalación de tus programas en tu iPhone.](#)

2010-03-04
[Sacar Release de un proyecto con Maven](#)

2010-03-03
[Instalación de Subversion y Apache en Ubuntu](#)

2010-03-03
[Cómo instalar la JDK de SUN en Fedora Linux](#)

2010-03-02
[Creando un botón de compra de Paypal con datos cifrados](#)

2010-03-01
[Creación de un plugin de tipo hook en Liferay](#)

2010-03-01
[ScrumCards de Autentia en Android](#)

2010-02-25
[Creando la baraja de SCRUM de Autentia como aplicación para Android](#)

2010-02-25
[Instalar CentOS en Virtualbox con NetInstall](#)

2010-02-22
[Expresiones CRON](#)

2010-02-19
[Cómo utilizar el DataStore de Google App Engine con JDO](#)

2010-02-19
[Recursos Freeware](#)

3.3. Uso de la plantilla.

Por último, las páginas que hagan uso de la plantilla tendrán un contenido similar al siguiente:

```
01 <?xml version="1.0" encoding="UTF-8"?>
02 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
03 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
04 <html xmlns="http://www.w3.org/1999/xhtml" xmlns:ui="http://java.sun.com/jsf/facelets"
05 xmlns:f="http://java.sun.com/jsf/core" xmlns:h="http://java.sun.com/jsf/html">
06
07 <ui:composition template="/WEB-INF/facelets/templates/defaultLayout.xhtml">
08   <ui:define name="title">
09     <h:outputText value="#{msg.edit_of} #{msg.menu_items}" />
10   </ui:define>
11   <ui:define name="content">
12     <h:form>
13       <!-- contenido de la página-->
14     </h:form>
15   </ui:define>
16 </ui:composition>
17 </html>
```

De este modo, en el único punto en el que se define la estructura del site es en la plantilla y el código de las páginas queda centrado en el contenido individual de cada una de ellas y no nos repetimos - Please DRY: Don't Repeat Yourself -.

4. Componentes por composición.

Los componentes por composición son un tipo especial de plantillas que actúan como componentes. Un componente por composición es una pieza de código reutilizable que proporciona cierta funcionalidad y consiste en una colección de otros componentes que ya existen.

Con facelets, cualquier página xhtml puede convertirse en un componente por composición, haciendo uso de las siguientes etiquetas:

- **composite:interface:** declara el contrato de uso del componente por composición.
- **composite:implementation:** define la implementación del componente por composición. Si existe un composite:interface, debe existir su composite:implementation correspondiente.
- **composite:attribute:** declara un atributo en el contrato del componente, en la interface. Es una etiqueta hija de composite:interface.
- **composite:insertChildren:** sustituye a la etiqueta ui:insert que antes usábamos en el componente para insertar el conjunto de nodos hijos que se puede declarar dentro del etiquetado que hace uso del componente por composición. Se incluye dentro de la etiqueta composite:implementation.
- **composite:valueHolder:** permite exponer las propiedades y eventos de los componentes que implementan la interfaz ValueHolder para que sean asignados desde el cliente que hace uso del componente por composición. Los componentes que implementan EditableValueHolder son todos aquellos que soportan el acceso a propiedades del lado de los ManagedBeans a través de Expression Language y la posibilidad de asignar a dicha vinculación un conversor (outputText, outputLabel, outputFormat,...)
- **composite:editableValueHolder:** permite exponer las propiedades y eventos de los componentes que implementan la interfaz EditableValueHolder para que sean asignados desde el cliente que hace uso del componente por composición. EditableValueHolder es una extensión de ValueHolder que permite asociar al componente que la implementa validadores y eventos de cambio de valor (inputText, inputSecret, selectOneMenu, selectManyMenu, selectBooleanCheckBox,...).
- **composite:actionSource:** permite exponer las propiedades y eventos de los componentes que implementan la interfaz ActionSource para que sean asignados desde el cliente que hace uso del componente por composición. Los componentes que implementan ActionSource son aquellos que soportan eventos de acción y listeners de eventos de acción (commandButton y commandLink).

4.1. Configuración.

JSF 2 deja abierta la posibilidad de configurar nuestros componentes por composición a través de la declaración de los mismos en una taglib e incluyéndola en el descriptor de despliegue web.xml. Si bien, además, haciendo uso del concepto de convención sobre configuración, nos hace transparente dicha configuración de modo que nos propone un prefijo de URI predefinida para hacer referencia a los mismos y el sufijo vendrá dado por el directorio en el que se encuentren los xhtml que implementan los componentes por composición.

Es tan sencillo, como crear nuestros componentes bajo una estructura de directorios components/autentia dentro de la carpeta de recursos (resources) para que, cuando se empaquete se incluya dentro del directorio WEB-INF o META-INF. De este modo todos los xhtml que contengan la declaración de componente serán incluidos en una librería de componentes cuya URI tendrá una parte fija (http://java.sun.com/jsf/composite/) y un sufijo que se corresponderá con la estructura de directorios en el que esté incluido (components/autentia). El nombre del componente será el nombre del fichero xhtml.

Así, podemos mantener la siguiente estructura de directorios:



e incluir el espacio de nombres de los mismos en nuestros xhtml de la siguiente forma:

```
1 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
2 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
3 <html xmlns="http://www.w3.org/1999/xhtml"
4       xmlns:tnt="http://java.sun.com/jsf/composite/components/autentia">
5 ...
```

4.2. Creación de un componente por composición.

Para definir un componente por composición en JSF 2, ahora se requiere la definición de la interfaz y la implementación. Qué mejor

2010-02-17
[Plugin de mejora de graficos para JMeter](#)

2010-02-17
[Cómo utilizar el datastore de Google App Engine con su API de nivel inferior](#)

2010-02-16
[Aprendiendo Objective-C desarrollando para nuestro Iphone 3GS](#)

2010-02-11
[Introducción a JCL.](#)

2010-02-09
[Creando la Baraja de SCRUM de Autentia como aplicación para el Iphone 3G.](#)

2010-02-08
[Cómo generar versiones imprimibles de páginas web](#)

2010-02-04
[Como cambiar el tamaño de las fuentes en Xcode \(el entorno de desarrollo para Mac e iPhone\)](#)

2010-02-04
[Primeros pasos con Enterprise Architect y UML 2.x](#)

2010-02-04
[Creación de un componente JSF, basádonos en un plugin de jQuery, con el soporte de RichFaces.](#)

2009-02-03
[Sincronizando el Mail de Mac con Gmail, el correo de Google](#)

2010-02-03
[Integración de jQuery en RichFaces.](#)

2010-02-02
[AjaxSingle: el partialSubmit de RichFaces.](#)

2010-02-01
[Introducción a RichFaces.](#)

2010-01-29
[Transformación de mensajes en SOA con OpenESB](#)

2010-01-26
[JMeter. Uso de funciones.](#)

2010-01-18
[Autenticando los usuarios de Sonar contra un LDAP](#)

2010-01-18
[Introducción a jQuery UI.](#)

2010-01-18
[jQuery: cómo crear nuestros propios plugins.](#)

que mostrar un componente complejo para entenderlo, se trata de un componente de entrada de datos que encapsula la etiqueta, el propio componente de entrada y el componente de mensajes asociado al anterior.

```
01 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
02 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
03 <html xmlns="http://www.w3.org/1999/xhtml"
04     xmlns:h="http://java.sun.com/jsf/html"
05     xmlns:f="http://java.sun.com/jsf/core"
06     xmlns:ui="http://java.sun.com/jsf/facelets"
07     xmlns:composite="http://java.sun.com/jsf/composite">
08 <h:head>
09
10     <title>This will not be present in rendered output</title>
11
12 </h:head>
13
14 <h:body>
15
16     <composite:interface displayName="Input text field with label, input and message components" >
17         <composite:attribute name="entity" required="true" />
18         <composite:attribute name="field" required="true" />
19         <composite:attribute name="label" required="false"
20 default="#{cc.attrs.entity.class.simpleName}.#{cc.attrs.field}" />
21         <composite:attribute name="required" required="false" default="false" />
22     </composite:interface>
23
24     <composite:implementation>
25
26         <h:panelGroup id="containerOfLabel">
27             <h:outputLabel id="label"
28 for="#{cc.attrs.field}" value="#{msg[cc.attrs.label]}" />
29             <ui:fragment rendered="#{cc.attrs.required}">
30                 <h:outputText value="(*)" styleClass="red"/>
31             </ui:fragment>
32             <h:outputText value=":" />
33         </h:panelGroup>
34
35         <h:inputText id="input" required="#{cc.attrs.required}"
36 value="#{cc.attrs.entity[cc.attrs.field]}" >
37             <f:ajax render="message" />
38         </h:inputText>
39
40         <h:message id="message" for="input" errorClass="errorMessages" />
41     </composite:implementation>
42 </h:body>
43 </html>
```

Ahora tenemos:

- por un lado la definición de la interfaz en la que se pueden definir los parámetros que recibirá o puede recibir el componente incluidos en la etiqueta composite:interface, esto es bastante útil porque eliminamos la necesidad de incluir tediosos bloques de <c:if comprobando el contenido de los parámetros.
- por otro lado, la propia implementación dentro de la etiqueta composite:implementation, que hará uso de los parámetros definidos en la interfaz: dentro de JSF 2 cc es una palabra reservada para hacer referencia al componente por composición y que permite acceder al valor de los atributos definidos en la interfaz.

Todo lo que no esté incluido dentro de estas dos etiquetas no será renderizado.

4.3. Uso de un componente por composición.

Para hacer uso de un componente por composición basta con uncluir el espacio de nombres con la URI que hemos comentado anteriormente:

```
01 <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
02 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
03 <html xmlns="http://www.w3.org/1999/xhtml"
04     xmlns:h="http://java.sun.com/jsf/html"
05     xmlns:f="http://java.sun.com/jsf/core"
06     xmlns:ui="http://java.sun.com/jsf/facelets"
07     xmlns:tnt="http://java.sun.com/jsf/composite/components/autentia">
08 <h:head>
09     <title>Composite component example...
10 </h:head>
11
12 <h:body>
13     <h:form>
14         <tnt:inputText entity="#{beanTestView.item}" field="reference" required="true"/>
15     </h:form>
16 </h:body>
17
18 </html>
```

4.4. Exponer las propiedades y comportamientos de los componentes de un componente por composición.

Adicionalmente podemos incluir dentro de la declaración de la interfaz una serie de etiquetas que exponen los componentes que implementan ciertas interfaces hacia el exterior, de modo que el cliente que usa el componente puede redefinir sus propiedades y comportamientos frente a eventos. En el siguiente ejemplo exponemos el componente de entrada de texto de nuestro componente por composición:

```
1 <composite:interface displayName="Input text field with label, input and message components" >
2     ...
3 <composite:editableValueHolder name="input" />
4     ...
5 </composite:interface>
```

De este modo, quien hace uso del componente puede asignar comportamientos típicos de la interfaz de implementa (convertidores, validadores, eventos de acción y de cambio de valor):

```
1 <tnt:inputText id="description" entity="#{beanTestView.item}" field="description">
```

2010-01-18
Cómo consumir un servicio web RESTful con el soporte de Ajax y JSON de JQuery.

2010-01-18
Introducción a JQuery.

2010-01-17
Introducción a Tapestry 5

2010-01-14
JMeter. Gestión de usuarios

2010-01-14
Patrón Visitor con commons-collections y sus Closures

2010-01-12
Creación de servicios web RestFul, con soporte a persistencia, en NetBeans.

2010-01-11
JMeter y JSF. Extracción del parámetro ViewState

2010-01-07
Importar el correo de Microsoft Outlook al cliente de correo de Mac OS.

2010-01-07
Monitor de Hudson para Eclipse.

2010-01-07
Patrones de diseño de XML Schema

Últimas ofertas de empleo

2009-07-31
T. Información - Operador (día / noche) - BARCELONA.

2009-06-25
Atención a cliente - Call Center - BARCELONA.

2009-06-19
Otras - Ingenieria (minas, puentes y puertos) - VALENCIA.

2009-06-17
Comercial - Ventas - ALICANTE.

2009-06-03
Comercial - Ventas - VIZCAYA.

Anuncios Google


```
2 <f:validateLength for="input" minimum="1" maximum="20" />  
3 </tnt:inputText>
```

Para hacer esto en JSF 1.2 usábamos la etiqueta `ui:insert` y ahora podríamos hacer uso de la etiqueta `composite:insertChildren`, si bien sólo podríamos asignar el contenido de la etiqueta a un componente dentro del componente por composición, ahora es más parametrizable y permite crear componentes por composición más complejos. Además, si pensamos en cómo parametrizamos en JSF 1.2 el método del controlador que recibirá un evento de cambio de valor y lo trasladamos a JSF 2, rápidamente veremos los beneficios.

En general, se acabaron las etiquetas condicionales porque, al exponer los componentes hacía fuera es el cliente que los usa quien define su comportamiento.

5. Referencias.

- <http://java.sun.com/javaee/javaxserverfaces/2.0/docs/pdldocs/facelets/index.html>
- <http://www.ibm.com/developerworks/java/library/j-jsf2fu2/index.html>
- http://blogs.sun.com/enterprisetechtips/entry/true_abstraction_composite_ui_components1

6. Conclusiones.

Cada vez nos gusta más lo que vamos encontrando... ;-)

Se produce además la paradoja de que la mayoría de las librerías de componentes visuales de terceros con las que trabajamos, que son además en las que se ha inspirado la especificación (el Ajax de JSF 2, esrá inspirado en el de RichFaces: Ajax4JSF), por la complejidad de sus componentes, aún no soportan JSF 2, lo cual nos está dando pie a evaluar otras librerías más recientes que sí lo soportan: véase [Primefaces](#).

Desde [Autentia](#), continuaremos compartiendo el conocimiento que vamos adquiriendo y, si estáis interesados en el contenido de nuestros tutoriales y tenéis una necesidad formativa al respecto, no dudeis en poneros en [contacto](#) con nosotros. En [Autentia](#) nos dedicamos, además de a la consultoría, desarrollo y soporte a desarrollo, a impartir [cursos de formación](#) de las tecnologías con las que trabajamos.

Un saludo.

Jose

jmsanchez@autentia.com

¿Qué te ha parecido el tutorial? Déjanos saber tu opinión y ivota!

Muy malo Malo Regular Bueno Muy bueno



Votar

(Sólo para usuarios registrados)

» [Regístrate y accede a esta y otras ventajas](#) «

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](#) «

Autor

Mensaje de usuario registrado

- Puedes inscribirte en nuestro servicio de notificaciones [haciendo clic aquí](#).
- Puedes firmar en nuestro libro de visitas [haciendo clic aquí](#).
- Puedes asociarte al grupo AdictosAlTrabajo en XING [haciendo clic aquí](#).
- Añadir a favoritos Technorati.  ADD THIS BLOG TO MY **Technorati FAVORITES**



Esta obra está licenciada bajo [licencia Creative Commons de Reconocimiento-No comercial-Sin obras derivadas 2.5](#)

Recuerda

Autentia te regala la mayoría del conocimiento aquí compartido ([Ver todos los tutoriales](#)). Somos expertos en: J2EE, Struts, JSF, C++, OOP, UML, UP, Patrones de diseño ... y muchas otras cosas.

¿Nos vas a tener en cuenta cuando necesites consultoría o formación en tu empresa?, ¿Vas a ser tan generoso con nosotros como lo tratamos de ser con vosotros?

Somos pocos, somos buenos, estamos motivados y nos gusta lo que hacemos ...

Autentia = Soporte a Desarrollo & Formación.

info@autentia.com

Tutoriales recomendados

Nombre	Resumen	Fecha	Visitas	Valoración	Votos	Pdf
Facelets en JSF 2: sistema de plantillas y componentes por composición.	Ahora que ya conocemos las principales novedades de JSF 2, en este tutorial vamos a realizar una introducción al uso de Facelets en JSF 2: cómo crear planillas y componentes por composición.	2010-04-20	9	-	-	
Introducción a RichFaces.	RichFaces es una librería de componentes visuales para JSF con soporte para Ajax4JSF.	2010-02-01	2784	-	-	
Tutorial basico de google wave bots	En este tutorial crearemos un robot muy simple, que al invitarlo a un Wave sustituye una tag por la fecha y hora actual.	2009-11-18	2774	-	-	
Migración de JSP a Facelets	Si tienes alguna aplicación que mantener basada en JSF y construida en JSPs, este tutorial te servirá para evaluar el coste de su migración a Facelets	2008-10-22	8346	Bueno	2	
Metro: pila de webservices de Sun. Integración con Maven 2	En este tutorial Germán nos enseñara a integrar la generación de webservices con Metro y Maven2.	2008-04-05	3563	-	-	
Jersey: la implemetación de RESTFull de Sun	En este tutorial Germán nos enseña cómo usar RESTFull con la tecnología de Sun.	2008-04-05	2846	-	-	
Genera gráficas como las de Google Chart con Eastwood	Si no conoces JFreeChart pero quieres hacer gráficas tan llamativas como las de Google Chart, Eastwood te resultará muy útil	2008-04-03	4293	-	-	
Introducción a JPivot	Nuestro compañero Juan nos enseña la librería JPivot que se utiliza para mostrar tablas dinámicas en Java	2008-03-04	11152	-	-	
Ejemplo de web con ICEfaces	Creación de una web paso a paso con ICEFaces, Tomcat 5.5 y Eclipse	2008-01-16	22489	-	-	
Integración de JSF 1.2, Facelets e ICEFaces en Tomcat 6	Integración de JSF 1.2, Facelets e ICEFaces en Tomcat 6	2007-12-10	15222	-	-	

Nota:

Los tutoriales mostrados en este Web tienen como objetivo la difusión del conocimiento. Los contenidos y comentarios de los tutoriales son responsabilidad de sus respectivos autores. En algún caso se puede hacer referencia a marcas o nombres cuya propiedad y derechos es de sus respectivos dueños. Si algún afectado desea que incorporemos alguna reseña específica, no tiene más que solicitarlo. Si alguien encuentra algún problema con la información publicada en este Web, rogamos que informe al administrador rcanales@adictosaltrabajo.com para su resolución.