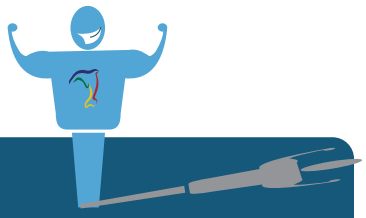
autentia

Avenida de Castilla,1 - Edificio Best Point - Oficina 21B
28830 San Fernando de Henares (Madrid)
tel./fax: +34 91 675 33 06
info@autentia.com - **www.autentia.com**

¿**Qué ofrece** Autentia Real Business Solutions S.L?

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

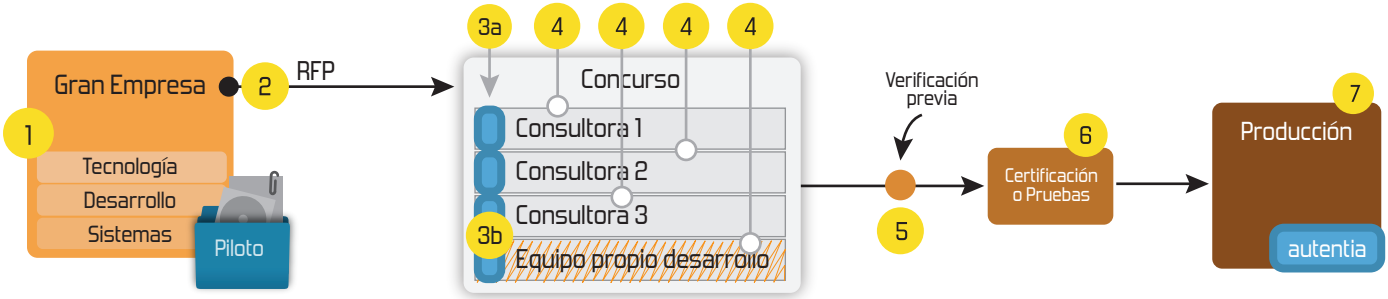
1. Desarrollo de componentes y proyectos a medida



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

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

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

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

Compartimos nuestro conocimiento en:
www.adictosaltrabajo.com

Para más información visítenos en:
www.autentia.com

NUEVO ¿Quieres saber cuánto ganas en relación al mercado? pincha aquí...

Ver cursos que ofrece Autentia

Descargar comics en PDF y alta resolución

[¡NUEVO!] 2008-12-01

2008-11-17

2008-09-01

2008-07-31

Estamos escribiendo un libro sobre la profesión informática y estas viñetas formarán parte de él. Puedes opinar en la seccion comic.

Tutorial desarrollado por

Jose Manuel Sánchez Suárez

Consultor tecnológico de desarrollo de proyectos informáticos. Diseñador de Adictos Al Trabajo 2.0

Puedes encontrarme en Autentia

Somos expertos en Java/J2EE

Catálogo de servicios de Autentia

Descargar (6,2 MB)

Descargar en versión comic (17 MB)

AdictosAlTrabajo.com es el Web de difusión de conocimiento de Autentia.

autentia real business solutions

Catálogo de cursos

Descargar este documento en formato PDF: friendlyurlrewritefilter.pdf

Fecha de creación del tutorial: 2008-12-17

URLs amigables con UrlRewriteFilter.

0. Índice de contenidos.

1. Introducción

Son URLs amigables aquellas entendibles por el usuario, intuitivas y formadas por palabras claves relacionadas con el contenido de la página que muestran.

Últimos tutoriales

El objetivo es evitar la construcción de enlaces a páginas dinámicas llenos de parámetros vía GET:

```
view plain  print  ?
01. http://www.miweb.com/libros/view.jsf?id=3&categoria=21
```

sustituyendo dicha sintaxis por otra más amigable:

```
view plain  print  ?
01. http://www.miweb.com/libros/3/21/Spring-in-Action
```

donde Spring in Action sería el título del libro que visualizamos.

Los motivos que nos llevan a dedicar un esfuerzo adicional en este sentido son, principalmente, dos:

- el hecho de tener URLs amigables hace que el usuario entienda la ubicación desde la propia barra de navegación, permitiendo crear bookmarks que no caduquen porque, por ejemplo, modifiquemos el nombre de los parámetros en un futuro,
- no hay que olvidar que existen otros consumidores de nuestro site que no son usuarios humanos ;). Los motores de indexación de los motores de búsqueda entienden mejor las URLs amigables, puesto que dan un mayor peso a la página en función de los literales de la URL. Las técnicas de optimización de web sites para motores de búsqueda (SEO: Search Engine Optimization), entre otras muchas recomendaciones, indican que si una URL tiene más de un parámetro no será indexada; esto puede variar en función del motor de búsqueda, pero si tenemos en cuenta que google, hasta hace poco, no indexaba una URL que tuviese un parámetro llamado "id", vemos el esfuerzo como una necesidad.

Ya hemos hablado en adictos sobre URLs amigables con Spring MVC, puesto que lo ideal es que el framework sobre el que desarrollamos, si implementa un patrón MVC, nos de una solución para construir este tipo de URLs.

Pero podemos no tener esa suerte, y necesitar de una implementación aparte que nos solucione nuestro problema de "URLS sucias".

La solución más limpia, si disponemos de la infraestructura necesaria, es delegar la traducción de "URL amigable" a "URL sucia" al servidor web. Apache Web Server tiene un módulo llamado mod_rewrite que, frente a una petición, es capaz de analizar la URL y sobreescribirla, de modo que a la aplicación web le llegue otra URL totalmente diferente. Así usamos el servidor web, como proxy que es, para realizar un mapeo entre "URL amigable" y "URL sucia", de modo que esta última sólo viva en el ámbito del servidor de aplicaciones.

Pero no siempre vamos a disponer de esta infraestructura o, disponiendo de ella, quizás no tengamos acceso a su configuración o, simplemente, no queramos depender de la gente de sistemas (sorry ;)).

Para ello, en cualquier aplicación J2EE, podemos hacer uso de un filtro en su módulo web, basado en el módulo mod_rewrite de Apache Web Server, que nos permite sobreescribir URLs antes de llegar a nuestro código.

Examinar el filtro es el objetivo de este tutorial y su nombre es UrlRewriteFilter.

Podemos hacer uso del UrlRewriteFilter para:

- hacer la conversión de "URL amigable" a "URL sucia" y viceversa: independientemente de la tecnología o framework con el que estemos trabajando: JSP, Servlets, Struts, JSF,...
- detectar el navegador del cliente, de modo que permite redireccionar una petición en función del user-agent,
- realizar una redirección programada, en función a una hora y/o fecha, ideal para realizar tareas de mantenimiento sobre el site,
- mover contenido de forma transparente para el cliente. Lo normal es que, si realizamos una reorganización del contenido de nuestro site y este ya ha sido indexado por algún motor de búsqueda, perdamos el peso de las URLs tras el cambio. De este modo podemos seguir manteniendo los enlaces anteriores hasta que los nuevos cobren el mismo peso.

Siempre que utilizamos una nueva tecnología tratamos de buscar buenas referencias. Sobre UrlRewriteFilter:

- le podemos encontrar en los repositorios públicos de Maven,
- está alojado bajo code.google.com, lo cuál da garantía de mantenibilidad,
- Jboss Seam se basa en ésta librería para realizar su conversión interna de URLs, de modo que no requiere de un fichero de configuración aparte, está integrado en el propio pages.xml

2. Entorno.

El tutorial está escrito usando el siguiente entorno:

- Hardware: Portátil Asus G1 (Core 2 Duo a 2.1 GHz, 2048 MB RAM, 120 GB HD).
- Sistema operativo: Windows Vista Ultimate.
- JDK 1.5.0_15
- Eclipse 3.4, con q4e.

3. Apache mod_rewrite.

Como hemos comentado, UrlRewriteFilter tiene la misma filosofía que el módulo de Apache Web Server mod_rewrite, de hecho es una implementación del mismo, pero como un filtro dentro de una aplicación J2EE.

El módulo mod_rewrite es una pequeña extensión de Apache que nos permite que, frente a una petición de una URL, realmente se esté llamando a otra URL totalmente diferente. Para ello utiliza un intérprete de expresiones regulares traduciendo URL's en tiempo de ejecución.

Acepta las configuraciones especificadas desde el archivo global httpd.conf, o configuraciones específicas de directorio utilizando el fichero (.htaccess).

El filtro UrlRewriteFilter se puede configurar en base a un fichero .htaccess del módulo mod_rewrite.

4. Maven: el pom.xml.

Como hemos comentado, podemos encontrar la librería en los repositorios públicos de Maven:

```
view plain  print  ?
01. <project ...>
02.     ...
03.   <dependency>
04.     <groupId>org.tuckey</groupId>
05.     <artifactId>urlrewritefilter</artifactId>
06.     <version>3.1.0</version>
07.   </dependency>
08.     ...
09. </project>
```

También podemos descargarla e incluirla en el directorio WEB-INF/lib si no disponemos de Maven.

5. Configuración.

Debemos añadir el filtro en la configuración del descriptor de despliegue **web.xml**, de nuestra aplicación web:

```
view plain  print  ?
01. ...
02.     <filter>
03.        <filter-name>UrlRewriteFilter</filter-name>
04.        <filter-class>org.tuckey.web.filters.urlrewrite.UrlRewriteFilter</filter-class>
05.         <!-- indica la cantidad de segundos que será usada para chequear si el fichero de
06.         y se requiere su recarga (-1 indica que no será recargado, usar en producción) --
07.         <init-param>
08.             <param-name>confReloadCheckInterval</param-name>
09.             <param-value>1</param-value>
10.         </init-param>
11.
12.         <!-- path del fichero de configuración (por defecto /WEB-INF/urlrewrite.xml) -->
13.         <init-param>
14.             <param-name>confPath</param-name>
15.             <param-value>/WEB-INF/urlrewrite.xml</param-value>
16.         </init-param>
17.
18.         <!-- nivel de log (por defecto WARN) -->
19.         <init-param>
20.             <param-name>logLevel</param-name>
21.             <param-value>DEBUG</param-value>
22.         </init-param>
23.
24.         <!-- path del monitor de estado del filtro, debe comenzar con / -->
25.         <init-param>
26.             <param-name>statusPath</param-name>
27.             <param-value>/rewriteStatus</param-value>
28.         </init-param>
29.
30.         <!-- indica si el monitor de estado está activo (por defecto true) -->
31.         <init-param>
32.             <param-name>statusEnabled</param-name>
33.             <param-value>true</param-value>
34.         </init-param>
35.
36.         <!-- hosts que tendrán acceso al monitor de estado, se puede usar el * como
37.         comodin (por defecto "localhost, local, 127.0.0.1") -->
38.         <init-param>
39.             <param-name>statusEnabledOnHosts</param-name>
40.             <param-value>localhost</param-value>
41.         </init-param>
42.
43.         <!-- por defecto false. Indica que se usará el estilo de configuración del módulo
44.         Si está activado buscará un fichero de configuración /WEB-INF/.htaccess (se puede
45.         <init-param>
46.             <param-name>modRewriteConf</param-name>
47.             <param-value>false</param-value>
48.         </init-param>
49.     </filter>
50.     <filter-mapping>
51.        <filter-name>UrlRewriteFilter</filter-name>
52.        <url-pattern>/*</url-pattern>
53.     </filter-mapping>
54. ...
```

6. Parametrización.

UrlRewriteFilter usa un fichero xml, por defecto /WEB-INF/urlrewrite.xml, en el que se definen reglas.

Los parámetros en los que se definen las URLs, pueden usar el estilo de

- expresiones regulares (por defecto). En las expresiones regulares ^ señala el inicio de una cadena y $ el final y los paréntesis permite la definición de variables accesibles vía $1, $2,...
- expresiones con comodines.

```
view plain  print  ?
01. <!DOCTYPE urlrewrite
02.     PUBLIC "-//tuckey.org//DTD UrlRewrite 3.0//EN"
03.     "http://tuckey.org/res/dtds/urlrewrite3.0.dtd">
04.
05. <urlrewrite>
06.
07.   <!-- /some/old/guide.pdf será sobreescrita a /very/new/guide.pdf -->
08.   <rule match-type="wildcard">
09.       <from>/some/old/*.pdf</from>
10.       <to type="redirect">/very/new/$1.pdf</to>
11.   </rule>
12.
13.   <!-- /world/unitedstates/newyork será sobreescrita a /world.jsp?country=unitedstates&ci
14.   <rule>
15.       <from>^/world/([a-z]+)/([a-z]+)$</from>
16.       <to>/world.jsp?country=$1&city=$2</to>
17.   </rule>
18.
19.   <!-- detección del navegador: si es una versión antigua de mozilla se redirije a /some/
20.   <rule>
21.       <condition name="user-agent">Mozilla/[1-4]</condition>
22.       <from>^/some/page\.html$</from>
23.       <to>/some/page-for-old-browsers.html</to>
24.   </rule>
25.
26.   <!-- redirección programada: los domingos, de 8 a 10 de la noche, las peticiones a la /
27.   <rule>
28.       <condition type="dayofweek" next="and">1</condition>
29.       <condition type="hourofday" operator="greaterorequal" next="and">20</condition>
30.       <condition type="hourofday" operator="lessorequal">22</condition>
31.       <from>^/products/$</from>
32.       <to>/products/sunday-specials.html</to>
33.   </rule>
34.
35.   <!-- redirección de seguridad: invoca al método de la request isUserInRole(...). Ideal
36.   <rule>
37.       <condition type="user-in-role" operator="notequal">admin</condition>
38.       <from>^/admin/(.*)$</from>
39.       <to>/go-away-please.html</to>
40.   </rule>
41.
42. </urlrewrite>
```

La definición de la regla incluye:

- en la etiqueta rule el atributo **match-type**: por defecto a regex (expresión regular) se puede configurar a wildcard,
- **from**: URL de origen,
- **to**: URL de destino. Permite el uso del atributo **type**, por defecto se realiza un forward, pero también podemos configurar un redirect, permanent-redirect,...
- **condition**: para añadir condiciones a la redirección. Se pueden anidar.

Para ampliar la información sobre los parámetros de configuración recomendamos revisar la documentación sobre UrlRewriteFilter y algunos de sus ejemplos.

7. Monitor de estado.

Si lo tenemos habilitado, y accedemos desde un host permitido, podemos visualizar el estado del filtro accediendo desde una URL parecida a esta: **http://localhost:8080/seo/rewriteStatus**, donde **seo** es el contexto de la aplicación web y **rewriteStatus** el path del monitor configurado en el web.xml.

El monitor nos informa del estado del filtro:

- cuál es el fichero de configuración, la última vez que se leyó y cuándo se realizará la próxima lectura,
- las reglas que tenemos activadas, y
- la información a nivel de debug sobre la petición realizada al monitor.

8. Ejemplo.

Vamos a mostrar un ejemplo en el entorno de una aplicación con JSF.

Tenemos un listado de noticias, representado por la página **/webapp/pages/news/list.jspx**, que hace uso de una plantilla de facelets:

view plain print ?

```
<jsp:root xmlns:jsp="http://java.sun.com/JSP/Page" xmlns:ui="http://java.sun.com/jsf/face
    xmlns:h="http://java.sun.com/jsf/html" xmlns:f="http://java.sun.com/jsf/core">

    <ui:composition template="/WEB-INF/facelets/template/defaultLayout.jspx">
        <ui:define name="title">Listado de noticias</ui:define>
        <ui:define name="content">
            <div>
                <div class="news">
                    <h4><a href="news/1/Tenis/Rafa-Nadal-Numero-1-atp">Rafa Nadal #1 ATP<
                    <p>Lorem Ipsum is simply dummy text of the printing and typesetting i
              the industry's standard dummy text ever since the 1500s...</p>
                </div>
                <div class="news">
                    <h4><a href="news/2/Liga-de-Futbol-Professional">Resultados de la jor
                    <p>when an unknown printer took a galley of type and scrambled it to
              not only five centuries, but also the leap into electronic typesetting, ...
                </div>
                <div class="news">
                    <h4><a href="news/3/2008/10/15/Caída-de-las-bolsas">Caida y repunte d
                    <p>remaining essentially unchanged. It was popularised in the 1960s w
              Lorem Ipsum passages, and more recently with desktop publishing software li
                </div>
                <div class="news">
                    <h4><a href="news/4">Otra noticia</a></h4>
                    <p>empty.</p>
                </div>
            </div>
        </ui:define>
    </ui:composition>

</jsp:root>
```

Observamos que los enlaces al detalle de las noticias quedan fuera de componentes JSF, son simples enlaces html puesto que no queremos pasar por una regla de navegación de JSF, sino por una regla de sobreescritura del filtro.

Configuramos el filtro, el fichero **urlrewrite.xml**, con la siguiente regla:

view plain print ?

```
<!DOCTYPE urlrewrite
    PUBLIC "-//tuckey.org//DTD UrlRewrite 3.0//EN"
    "http://tuckey.org/res/dtds/urlrewrite3.0.dtd">

<urlrewrite>

    <rule>
        <from>^/news/([0-9]+)/(.*?)</from>
        <to>/pages/news/detail.jsf?id=$1&desc=$2</to>
    </rule>

</urlrewrite>
```

El enlace traducido por el filtro nos llevará a una página de detalle representada por la página **/webapp/pages/news/detail.jspx**, que hace uso de la misma plantilla de facelets:

view plain print ?

```
<jsp:root xmlns:jsp="http://java.sun.com/JSP/Page" xmlns:ui="http://java.sun.com/jsf/face
    xmlns:h="http://java.sun.com/jsf/html" xmlns:f="http://java.sun.com/jsf/core">

    <ui:composition template="/WEB-INF/facelets/template/defaultLayout.jspx">
        <ui:define name="title">Listado de noticias</ui:define>
        <ui:define name="content">
            <h:form>
                <p>Usted está en el detalle de la noticia #<h:outputText value="#{newsCtr
            </h:form>
        </ui:define>
    </ui:composition>

</jsp:root>
```

La página de detalle no es más que un ejemplo que se limita a imprimir los parámetro del identificador y la descripción de la noticia. Para ello se apoya de un managed bean que se ocupa de recuperar los valores de la request:

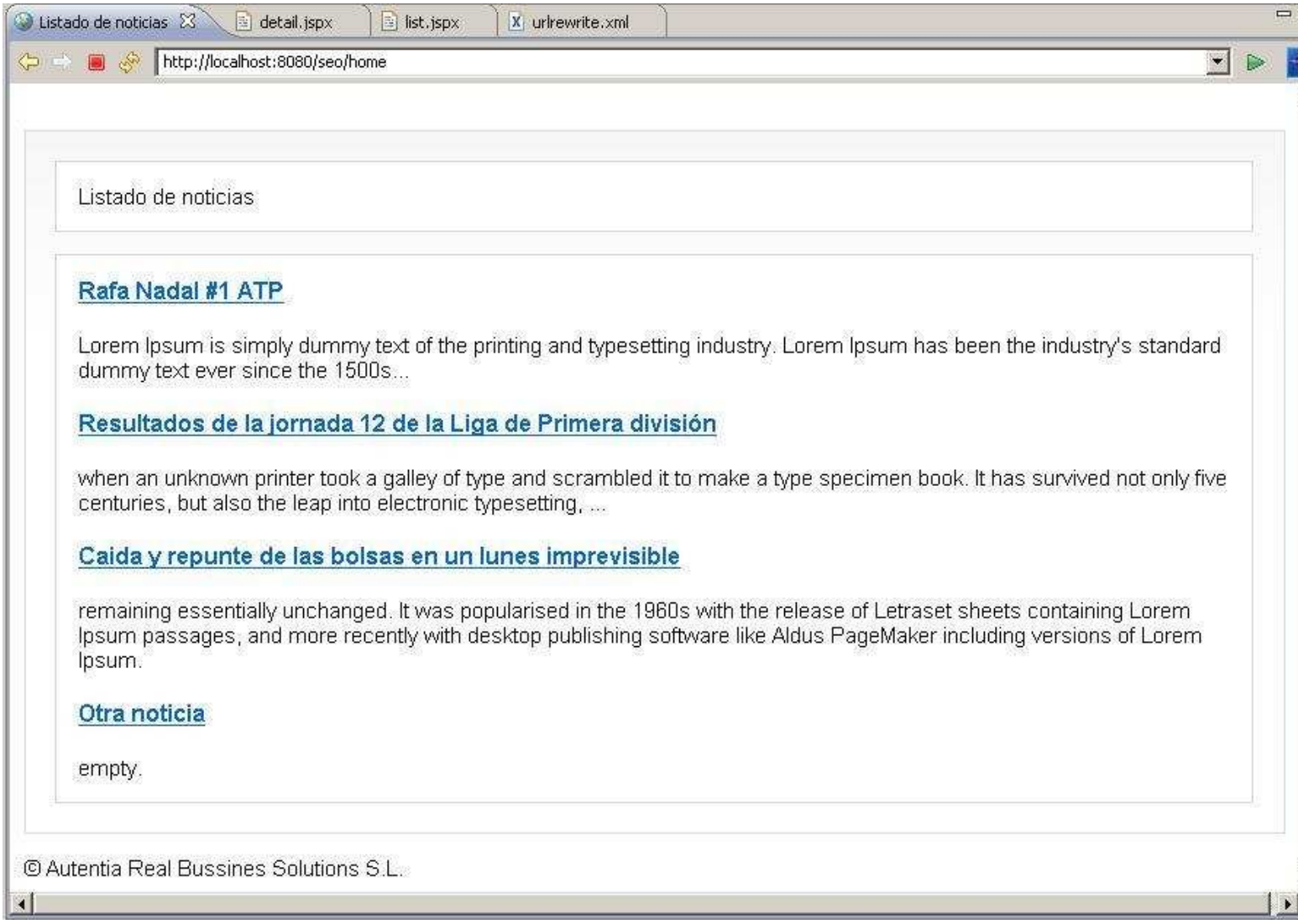
view plain print ?

```
01. package com.autentia.seo;
02.
03. import javax.faces.context.FacesContext;
04.
05. import org.springframework.stereotype.Controller;
06.
07. @Controller
08. public class NewsCtrl {
09.
10.     public String getId(){
11.         return getParameter("id");
12.     }
13.
14.     public String getDescription(){
15.         return getParameter("desc");
16.     }
17.
18.     private String getParameter(String name){
19.         try {
20.             return FacesContext.getCurrentInstance().getExternalContext().getRequestParam
21.         }
22.         catch (Exception e){
23.             e.printStackTrace();
24.         }
25.         return null;
26.     }
27. }
```

El método getParameter debería ir a una clase de utilidades!.

El resultado del ejemplo queda como sigue:

Listado de noticias



Si pulsamos sobre el título de la noticia nos lleva al detalle de la misma.

Detalle de una noticia

Podeis comprobar cómo la barra de direcciones mantiene una URL amigable, en la que, de sólo de un vistazo, sabemos que estamos consultando una noticia sobre "tenis" y en detalle sobre el "Número 1 de Rafa Nadal en la ATP", aunque dicha descripción a efectos de lógica de control la desechemos.

9. Conclusiones.

Si vuestro framework no os proporciona una solución para el manejo de URLs y no podéis hacer uso del módulo de Apache Web Server, mod_rewrite, con este filtro podemos emularlo a nivel de aplicación web.

Un saludo.

Jose

mailto:jmsanchez@autentia.com

- Puedes opinar sobre este tutorial 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

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

Servicio de notificaciones:

Si deseas que te enviemos un correo electrónico cuando introduzcamos nuevos tutoriales.

Formulario de subcripción a novedades:

E-mail [ ] Aceptar

Tutoriales recomendados

Nombre	Resumen	Fecha	Visitas	pdf
Apache Commons Configuration	En este tutorial os vamos a enseñar a utilizar una API de Apache para gestionar las configuraciones de vuestras aplicaciones de manera avanzada	2006-06-29	5090	pdf
Apache Jakarta Commons IO	En este tutorial vamos a hacer una presentación de la librería Jakarta Commons IO de Apache, que nos proporciona una extensión a la funcionalidad de los paquetes de Entrada/Salida que nos proporciona la plataforma J2SE.	2007-02-09	5983	pdf
Apache, MySQL y PHP	Os mostramos como configurar Apache, MySQL y PHP en vuestra máquina	2003-12-27	41080	pdf
Instalación de Apache Geronimo en Windows, e Integración con Eclipse Europa	En el siguiente tutorial os enseñaremos como instalar e integrar el servidor de aplicaciones J2EE Apache Geronimo. Nosotros nos basaremos en la versión 2.0.2 que está certificada J2EE 5.0.	2007-11-26	2619	pdf
Apache Commons Lang	En este tutorial se verá un ejemplo que inlcuyen clases y métodos interesantes.	2006-11-02	3865	pdf
Activación de la seguridad en Apache	Alejandro Pérez nos enseña como securizar Apache a través de autentificación básica y certificados de seguridad SSL.	2004-04-07	34185	pdf
mod_jk en WindowsXP / Apache2-JBoss	Os mostramos como instalar el conector mod_jk sobre WindowsXP utilizando Apache2 y JBoss	2005-12-30	7360	pdf
URLs amigables con Spring MVC	En este tutorial se va a hacer un ejemplo práctico utilizando Spring MVC para la configuración de URLs amigables de nuestra aplicación	2007-04-11	8064	pdf
Manual Básico de Apache iBatis	En este tutorial aprenderemos el uso básico de iBatis	2006-09-07	14981	pdf
Apache, Módulo JK (mod_jk) y JBoss	En este tutorial os vamos a enseñar a instalar un servidor apache, instalar el módulo JK para comunicar con dos servidores JBoss en cluster, para conseguir balanceo de carga y replicación de sesión .	2007-03-28	9371	pdf

Nota: