Generalitat de Catalunya
Centre de Telecomunicacions
i Tecnologies de la Informació

Framework Corporatiu J2EE

Servei de rewriting

Versió 1.0

Barcelona, 12 / setembre / 2006

Històric de modificacions

Data	Autor	Comentaris	Versió
04/09/2006	Atos Origin, sae openTrends	Versió inicial del document	1.0

Llegenda de Marcadors

Índex

1. Introducció

1.1. Propòsit

Aquest servei permet afegir de manera automàtica un prefix a les nostres URLs per tal de diferenciar entre peticions de recursos estàtics i peticions de recursos dinàmics, tal i com recomana el document Gestió de la Qualitat a l'apartat de convencions per a les Urls.

De la mateixa manera, aquest servei permet modificar les urls d'entrada a l'aplicació per tal de redirigir-les a un nou recurs.

Aquest servei està pensat per a ser utilitzat en un entorn en el qual es disposi d'un Servidor Web encarregat de servir les peticions de recursos estàtics (imatges, planes html,…) i d'un Servidor d'Aplicacions encarregat de servir les peticions de recursos dinàmics.

Igualment, i encara que el seu ús no és necessari, aquest servei també es pot fer servir en un entorn de desenvolupament en el qual no es disposi d'aquesta configuració. D'aquesta manera no cal diferenciar entre entorn de desenvolupament i entorn d'integració.

1.2. Versions i Dependències

En el present apartat es mostren quines són les versions i dependències necessàries per fer ús del Servei.

Nom	Tipus	Versió	Descripció
openFrame-services-web	jar	1.0.2	
urlrewrite	jar	3.0	http://tuckey.org/urlrewrite

1.3. A qui va dirigit

Aquest document va dirigit als següents perfils:

- Programador. Per conèixer l'ús del servei
- Arquitecte. Per conèixer quins són els components i la configuració del servei

1.4. Glossari

2. Descripció Detallada

2.1. Instal·lació i Configuració

2.1.1. Instal·lació

La instal·lació d'aquest servei només requereix la configuració d'un nou filtre web a la nostra aplicació, i la utilització de les dependències indicades a l'apartat 'Introducció-Versions i Dependències'.

2.1.2. Configuració

La configuració del Servei implica els següents pasos:

1) Configuració d'un filtre de l'aplicació Web.
2) Definir la configuració del fitxer de propietats del servei.
3) Configurar el servei de tags del framework.
4) Configurar el servei de seguretat.

Si el nostre projecte openFrame l'hem creat a partir del prototip de projecte del framework, tota aquesta configuració ja la tindrem feta, i l'únic que caldrà fer serà activar el filtre (en cas de que estigui comentat).

Definició de la implementació del filtre

Fitxer de configuració: web.xml

Ubicació proposada: <PROJECT_ROOT>/src/main/webapp/WEB-INF/web.xml

Per tal de fer servir el filtre s'ha d'afegir el següent codi al fitxer *web.xml*

```
<!-- Filter for rewriting URL -->
<filter>
       <filter-name>UrlRewriteFilter</filter-name>
<filter-
class>net.opentrends.openframe.services.web.filter.urlrewrite.UrlRewriteFilter</filter-
class>❶
   <init-param>
           <param-name>logLevel</param-name>
           <param-value>LOG4J</param-value>
   </init-param>
</filter>

….

<!-- Uncomment this filter mapping if Rewriting filter  has not been configured -->
<filter-mapping> ❷
  <filter-name>UrlRewriteFilter</filter-name>
  <url-pattern>/*</url-pattern>
</filter-mapping>
```

Quan es generen javascripts dinàmicament si emprem el servei URLRewritting aquests es generen sota /dwr, per això cal indicar el paràmetre interface sota el servlet de DWR per tal que aquests javascripts es generin sota l'AppJava, de manera que el servidor d'aplicacions els agafi correctament:

```
<!--  DWR SERVLET -->

<servlet>
        <servlet-name>dwr-invoker</servlet-name>
        <display-name>DWR Servlet</display-name>
        <servlet-class>uk.ltd.getahead.dwr.DWRServlet</servlet-class>

        <init-param>
                <param-name>debug</param-name>
                <param-value>true</param-value>
        </init-param>

        <init-param>
                <param-name>config</param-name>
                <param-value>/WEB-INF/classes/dwr/dwr.xml</param-value>
        </init-param>

        <init-param>
                 <param-name>interface</param-name>
                 <param-value>
                net.opentrends.openframe.services.web.dwr.impl.UrlRewriteInterfaceProcessor
                </param-value>
        </init-param>
</servlet>
```

Per a més informació es pot consultar la pàgina
http://tuckey.org/urlrewrite/manual/2.6/#filterparams

❶ És important fer servir la el filtre URLRewriteFilter del servei web del framework i no directament el proporcionat per la llibreria urlrewrite.jar.

La raó és que el filtre del servei web està modificat per tal de que el filtre pugui ser integrat dintre d'una aplicació en la qual també es trobin altres filtres que puguin modificar la url d'entrada, com pot ser el filtre de seguretat d'Acegi.

A l'apartat de preguntes freqüents, al final del document, podem trobar quin és el resultat de fer servir directament el filtre de la llibreria urlrewrite en una aplicació que també té configurat el filtre de seguretat d'Acegi.

❷ En una aplicació en la qual tenim configurats més d'un filtre, l'ordre en el qual s'apliquen depèn de l'ordre en el qual tinguem definits el <filter-mapping> al fitxer *web.xml*.

Per tant, per tal de deixar que el filtre de seguretat sigui sempre el primer en aplicar-se, hem de definir el filter-mapping per al nostre filtre al darrer lloc.

Definició de la configuració del Fitxer de Propietats del Servei

Fitxer de configuració: urlrewrite.xml

Ubicació proposada: <PROJECT_ROOT>/src/main/webapp/WEB-INF/urlrewrite.xml

La configuració d'aquest servei es fa mitjançant un fitxer xml, el qual s'ha de dir urlrewrite.xml i ha d'estar al directori WEB-INF de la nostra aplicació.

Podem trobar més informació sobre aquest fitxer a la pàgina següent:
http://tuckey.org/urlrewrite/manual/2.6/#configuration

Com a model de configuració d'una aplicació openFrame que vulgui assolir els requeriments marcats pel document Gestió de la Qualitat, al seu apartat Convencions per a les Urls, es proposa el següent fitxer:

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE urlrewrite PUBLIC "-//tuckey.org//DTD UrlRewrite 2.6//EN"
        "http://tuckey.org/res/dtds/urlrewrite2.6.dtd">

<!--

    Configuration file for UrlRewriteFilter
    http://tuckey.org/urlrewrite/

-->
<urlrewrite>

       <rule>
        <note>
            The rule means that request to .*/AppJava/.* (i.e. all request with AppJava
prefix)that not matches
            pattern .*/AppJava/dwr.*, will be redirected to a new resource with the same
request without
            AppJava prefix.
        </note>
              <condition type="request-url" operator="equal">.*/AppJava/.*</condition>
              <condition type="request-url"
operator="notequal">.*/AppJava/dwr.*</condition>
        <from>^(.*)/AppJava/(.*)$</from>
```

```
        <to last="true">$1/$2</to>
    </rule>

       <outbound-rule>
        <note>
        La outbound-rule especifica que quan s'invoca a response.encodeURL (o c:url si
        s'està fent servir JSTL) la url
           /*/AppJava/*.do*
        serà reescrita a
           /*/AppJava/*.do*.
        És a dir, es queda igual. Aquesta regla està pensada per a no modificar aquelles
        URLs de recursos dinàmics que ja tenen fixat el prefix AppJava.

        Per exemple, si la petició d'entrada és
            /openFrame-formacio/AppJava/home.do?reqCode=search
        la url reescrita serà la mateixa
            /openFrame-formacio/AppJava/home.do?reqCode=search

        D'altra banda, el fet de que l'atribut last estigui fixat a true fa que si aquesta
        regla s'aplica, ja no es miri la resta de reglas definides. Per tant, és important
        que aquesta sigui la primera regla a validar.
        </note>
        <from>^(.*)/AppJava/(.*)\.do(\Q?\E*.*)$</from>
         <to last="true">$1/AppJava/$2.do$3</to>
    </outbound-rule>

    <outbound-rule>
        <note>
        La outbound-rule especifica que quan s'invoca a response.encodeURL (o c:url si
        s'està fent servir JSTL) la url
           /*/*.do*
        serà reescrita a
           /*/AppJava/*.do*.
        És a dir, s'afegeix el prefix AppJava.

        Per exemple, si la petició d'entrada és
           /openFrame-formacio/admin/home.do?reqCode=search
        la url reescrita serà
           /openFrame-formacio/AppJava/admin/home.do?reqCode=search
        </note>
        <from>^/([^/]*)/(.*)\.do(\Q?\E*.*)$</from>
         <to last="true">/$1/AppJava/$2.do$3</to>
    </outbound-rule>

    <outbound-rule>
       <note>
       La outbound-rule especifica que quan s'invoca a response.encodeURL (o c:url si
       s'està fent servir JSTL) la url
             /*/AppJava/dwr/*
       serà deixada igual. Aquesta regla està pensada per a no modificar aquelles URLs
       de recursos gestionats pel servlet de DWR que ja tenen fixat el prefix AppJava.

       Per exemple, si la petició d'entrada és
             /openFrame-formacio/AppJava/dwr/interface/cuentas.js
       la url reescrita serà la mateixa
             /openFrame-formacio/AppJava/dwr/interface/cuentas.js

       D'altra banda, el fet de que l'atribut last estigui fixat a true fa que si aquesta
       regla s'aplica, ja no es miri la resta de reglas definides. Per tant, és important
       que aquesta sigui la primera regla a validar en el cas de peticions de recursos de
       DWR.
       </note>
        <from>^(.*)/AppJava/dwr/(.*)$</from>
           <to last="true">$1/AppJava/dwr/$2</to>
    </outbound-rule>

    <outbound-rule>
      <note>
        La outbound-rule especifica que quan s'invoca a response.encodeURL (o c:url si
```

```
        s'està fent servir JSTL) la url
            /*/dwr/*
        serà reescrita a
            /*/AppJava/dwr/*.

        Per exemple, si la petició d'entrada és
            /openFrame-formacio/dwr/interface/cuentas.js
        la url reescrita serà
            /openFrame-formacio/AppJava/dwr/interface/cuentas.js
         </note>
         <from>^(.*)/dwr/(.*)$</from>
            <to last="true">$1/AppJava/dwr/$2</to>
    </outbound-rule>

    <outbound-rule>
        <note>
        La outbound-rule especifica que quan s'invoca a response.encodeURL (o c:url si
        s'està fent servir JSTL) la url
              /*/j_acegi_security_check
        serà reescrita a
           /*/AppJava/j_acegi_security_check.
         </note>
         <from>^(.*)/j_acegi_security_check$</from>
            <to last="true">$1/AppJava/j_acegi_security_check</to>
    </outbound-rule>
</urlrewrite>
```

En aquest fitxer podem veure que tenim definides 1 regla per a les peticions d'entrada i 4 per a les peticions de sortida.

L'objectiu de les regles de sortida és afegir el prefix AppJava a totes aquelles urls que estiguin associades a recursos dinàmics (peticions que s'han de resoldre al servidor d'aplicacions).

En aquest cas, s'han identificat com a peticions de recursos dinàmics totes aquelles que segueixen el següent patró:

- *.do: peticions associades al controlador de Struts.
- */AppJava/dwr/* : peticions associades al servlet DWR (el prefix *AppJava/dwr* amb el qual s'identifiquen aquestes crides es configura al fitxer *web.xml*).
- */j_acegi_security_check: aquesta petició esta associada a l'acció que s'executa quan estem avaluant el formulari d'autentificació d'un usuari.

D'altra banda, si l'estructura de directoris de la nostra aplicació al servidor d'aplicacions no contempla l'ús de cap prefix, serà necessari que tota petició que arribi al servidor, abans de que arribi al DispatcherServlet, sigui modificada per tal de treure-li el prefix. Aquesta és la finalitat de la regla definida per a les peticions d'entrada.

Configurar el servei de tags del framework

Fitxer de configuració: openFrame-services-web.xml

Ubicació proposada: <PROJECT_ROOT>/src/main/resources/spring/openFrame-services-web.xml

En aquest fitxer hem d'activar l'atribut useUrlRewrite del bean ConfigurationTag:

```
<bean id="configurationTag"
class="net.opentrends.openframe.services.web.struts.taglib.configuration.
ConfigurationTag">
            <property name="styleId" value="defaultConfiguration"/>
            <property name="i18nService" ref="i18nService"/>
            <property name="appendContextPath" value="true"/>
            <property name="contextSubpath" value="AppJava"/>
            <property name="useUrlRewrite" value="true"/>
</bean>
```

D'aquesta manera li estem indicant al component del framework que genera les urls de manera dinàmica si ha de fer servir el servei de rewrite o no.

Configurar el servei de seguretat

Fitxer de configuració: openFrame-services-security.xml

Ubicació proposada: <PROJECT_ROOT>/src/main/resources/spring/openFrame-services-security.xml

En aquest fitxer, hem d'assegurar-nos de que tinguem les següents propietats al bean d'autentificació:

```
<property name="filterProcessesUrl" value="/AppJava/j_acegi_security_check"/>
<property name="loginFormUrlValue" value="/AppJava/pagelogin.do" />
<property name="authenticationFailureUrlValue" value="/AppJava/pagelogin.do" />
```

La raó per la qual s'ha de ficar el prefix *AppJava* a les dos darreres propietats és perquè el servei de seguretat el que fa és redirigir tota petició que s'hagi d'autentificar cap a l'acció indicada al paràmetre *loginFormUrlValue*. Per tant, si volem que aquesta petició sigui gestionada pel servidor d'aplicacions, haurà de tenir el prefix *AppJava*. Per la mateixa raó, en cas d'error en el procés d'autentificació, l'acció associada al paràmetre *authenticationFailureUrlValue* també ha de tenir el prefix.

I la raó per la qual s'ha de ficar el prefix *AppJava* a la primera propietat és perquè aquest paràmetre és el que permet identificar al servei de seguretat quina és l'acció associada al formulari d'autentificació.

2.2. Utilització del Servei

La utilització del servei comporta 2 passos:

1) Instal·lar i configurar el servei.
2) Definir la nostra pàgina JSP.

2.2.1. Instal·lar i configurar el servei

Només s'han de seguir els passos definits en el punt anterior.

2.2.2. Definir la nostra pàgina JSP

A les nostres JSP la única condició que hem de verificar per tal de que el servei funcioni és la que ja es comenta al propi document de *Gestió de la Qualitat*: : usar el tag <c:url> de JSTL per indicar la url de referència..

Tal i com es comenta al document, aquest tag afegeix de forma automàtica el context actual, fa rewriting URL per la gestió de sessions i s'encarrega de fer el encoding de tots els paràmetres i valors.

S'ha d'evitar l'ús de <%=request.getContextPath()%> i usar aquest tag.

A continuació es mostren uns exemples de construcció de urls amb aquest tag.

```
<link rel="stylesheet" href='<c:url value="/css/calendars/dynarch/calendar-system.css"/>'
type="text/css"/>
```

```
<form action='<c:url value="/j_acegi_security_check"/>' method="post">
```

El primer exemple correspon a una url que fa referència a un recurs estàtic. La segona petició fa referència a un recurs dinàmic.

Si ens fixem, en el moment de desenvolupar no cal fer aquesta distinció. Serà les regles definides al fitxer *urlrewriting.xml* les que s'encarregaran de fer la distinció.

Observació:
A les urls o accions definides amb els tags del framework no cal fer servir el tag <c:url>, ja que internament aquests tags fan servir la crida a *response.encodeUrl(),* que és la crida que fa servir el tag <c:url>.

Per exemple:

```
<fwk:form action="editAccount.do" styleId="actionForm" reqCode="edit" >
```

2.3. Integració amb Altres Serveis

2.3.1. Integració amb el Servei de Seguretat

Com ja s'ha comentat anteriorment, el servei de seguretat també fa servir un filtre que pot arribar a fer un rewriting de la url d'entrada.

Degut a que una petició d'entrada només pot ser redirigida cap a un nou recurs una sola vegada, en el cas de que el servei de seguretat faci un redirect cap a un nou recurs, el filtre de l'UrlRewrite no hauria de fer res.

Per a aconseguir aquest objectiu s'ha de garantir que el filter-maping del filtre de UrlRewrite estigui definit dintre del fitxer *web.xml* a posteriori del filter-mapping del servei de seguretat, com ja s'ha comentat en l'apartat de '*Instal·lació i Configuració*'.

2.4. Preguntes Freqüents

2.4.1. L'aplicació retorna un java.lang.IllegalStateException

Aquesta excepció es produeix quan sobre una request s'intenta fer un forward o un redirect i la petició ja ha sigut redirigida cap a un nou recurs en un pas anterior (es pot mirar l'atribut commit de la response associada a la request: si val false significa que encara no ha sigut redirigida i si val true és que si).

Hem d'assegurar-nos de dues coses:

1. Que el filter-mapping per al filtre UrlRewrite està després del filter-mapping del filtre de seguretat.
2. Que estem fent servir el filtre d'UrlRewrite del servei web i no el de la llibreria urlrewrite.

En aquest darrer cas, la única diferència entre els dos filtres radica en aquest fet: el filtre del servei web no fa res amb la request, i la envia cap al següent filtre de la cadena, si aquesta verifica que l'atribut commit de la response val true.