

# LIFERAY

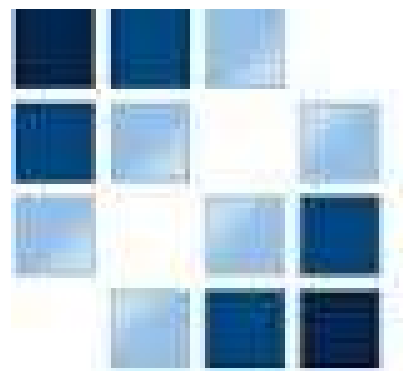
Enterprise. Open Source. For Life.

Desarrollo de Hooks en Liferay



# Contenidos

1. Introducción
2. Estructura del Proyecto
3. Hook de propiedades
4. Hook de Idioma
5. Hook de Action de Struts
6. Hook de JSP
7. Hook se Servicios



**LIFERAY**  
Enterprise. Open Source. For Life.

## INTRODUCCIÓN

# Introducción

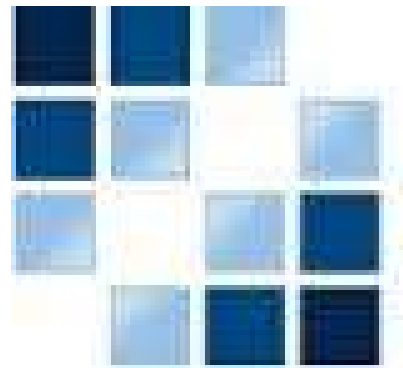
- Permiten "enganchar" y sobrescribir propiedades y funcionalidad del portal.
- Los hook, se configuran en el fichero **liferay-hook.xml**

# Introducción

- Los tipos de Hook posibles son
  - De propiedades del Portal.
  - De Idioma.
  - De Action de Struts.
  - De Vistas (JSP).
  - De Servicios.
  - De Indexer Post Procesor.

# Introducción

- Los **Hook**, permiten realizar una tarea semejante a la que permiten los plugin de tipo **Ext**.
- Los **Hook** son mas sencillo y mas fáciles de implementar que los **Ext**.
- La forma recomendada de extender o modificar el comportamiento del portal son los **Hook**.
- Solamente se emplearán los **Ext**, cuando los Hook no permitan la modificación pretendida.
- Los proyectos de tipo Hook, como el resto se pueden crear empleando **ANT** o **MAVEN**.



**LIFERAY**  
Enterprise. Open Source. For Life.

## ESTRUCTURA DEL PROYECTO

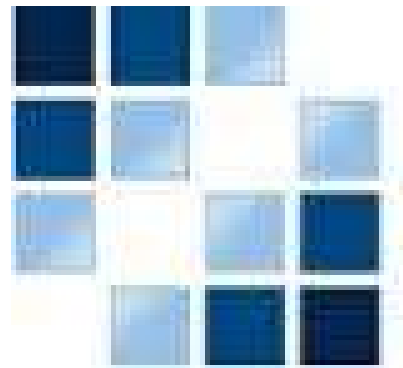
# Estructura del Proyecto

- Estructura del proyecto **ANT**
  - Proyecto/
    - docroot/WEB-INF/src/
    - docroot/
      - META-INF/
        - MANIFEST.MF
      - WEB-INF/
        - lib/
        - liferay-hook.xml
        - liferay-plugin-package.properties
        - web.xml
  - build.xml



# Estructura del Proyecto

- Estructura del proyecto **MAVEN**
  - Proyecto/
    - src/main/java
    - Src/main/webapp/
      - WEB-INF/
        - liferay-hook.xml
        - liferay-plugin-package.properties
        - web.xml
  - pom.xml



**LIFERAY**  
Enterprise. Open Source. For Life.

HOOK DE PROPIEDADES

# Hooks de propiedades

- Reemplaza propiedades
- Permite responder a eventos
  - Login de usuario
  - Logout de usuario
  - Ejecución de un servicio
  - Inicio de aplicación
  - Session de usuario
- Entrada en el liferay-hook.xml
  - <portal-properties>

# Hooks de propiedades

- No todas las propiedades pueden ser sobreescritas.
- Por ejemplo no podríamos sobrecribir las propiedades de la conexión a base de datos, eso hay que hacerlo con Ext.
- El listado de las propiedades que se pueden modificar se encuentra en el fichero

[https://docs.liferay.com/portal/6.2/definitions/liferay-hook\\_6\\_2\\_0.dtd.html](https://docs.liferay.com/portal/6.2/definitions/liferay-hook_6_2_0.dtd.html)

- Que es el esquema que sigue el fichero **liferay-hook.xml**

# Hooks de propiedades

- Las propiedades del **portal.properties** asociadas a estos eventos son
  - Login
    - login.events.pre
    - login.events.post
  - Logout
    - logout.events.pre
    - logout.events.post
  - Inicio de aplicación
    - application.startup.events

# Hooks de propiedades

- Las propiedades del **portal.properties** asociadas a estos eventos son
  - Ejecución de servicio
    - `servlet.service.events.pre`
    - `servlet.service.events.post`
  - Session de usuario
    - `servlet.session.destroy.events`
    - `servlet.session.create.events`

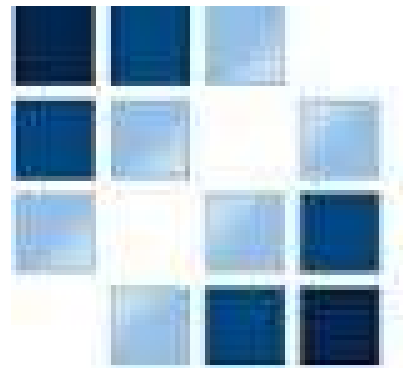
# Hooks de propiedades

- Estas propiedades admiten varios valores separados por comas, siendo los valores que se les han de asociar, los nombres de las clases que han de manejar el evento producido.
- Los eventos de **Login**, **Logout** y de **Servicio**, han de ser tratados por clases que extiendan de
  - **com.liferay.portal.kernel.events.Action**

# Hooks de propiedades

- Los eventos de **Inicio de Aplicación**, han de ser tratados por clases que extiendan de
  - **com.liferay.portal.kernel.events.SimpleAction**
- Los eventos de Session, por clases que extiendan
  - **com.liferay.portal.kernel.events.SessionAction**





**LIFERAY**  
Enterprise. Open Source. For Life.

HOOK DE IDIOMA

# Hooks de Idioma

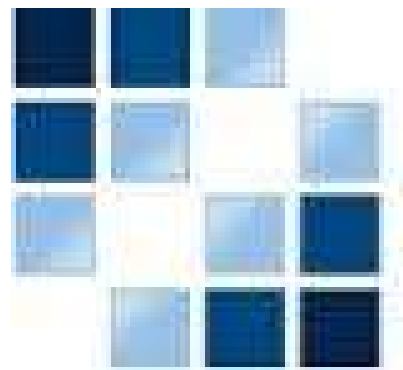
- Es un caso particular de Hook de propiedades
- Permite reemplazar y extender los ficheros de localización.
- Entradas en el **liferay-hook.xml**

```
<language-properties>
```

- Indicando un fichero de propiedades con el código idiomático y las traducciones correspondientes para las propiedades deseadas.

# Hooks de Idioma

- Se puede obtener un fichero **Language.properties** de referencia en la carpeta **content**, del fichero **portal-impl.jar**, que esta en la carpeta **WEB-INF/lib** del proyecto **ROOT** (el portal)



**LIFERAY**  
Enterprise. Open Source. For Life.

HOOK DE ACTION DE STRUTS

# Hook de Action de Struts

- En Liferay existen muchas funcionalidades accesibles creadas a través de Action de Struts.
- A través de los Hook, se pueden modificar estos Action o añadir nuevos.
- La configuración de estos Action se encuentra en el fichero

```
/portal-web/docroot/WEB-INF/struts-config.xml
```

# Hook de Action de Struts

- Para modificar un Action ya existente, se ha de definir una nueva entrada en el fichero **liferay-hook.xml**, indicando
  - El path del Action
  - La nueva clase que lo implemente

```
<hook>
  <struts-action>
    <struts-action-path>/login/login</struts-action-path>
    <struts-action-impl>
      com.ejemplo.actions.CustomLoginAction
    </struts-action-impl>
  </struts-action>
</hook>
```

# Hook de Action de Struts

- Para añadir un nuevo Action a parte de realizar lo anterior en el fichero **liferay-hook.xml**, se ha de incluir la nueva URL como path autorizado, indicando en el fichero **portal.properties** lo siguiente.

```
auth.public.paths=/portal/miejemplo
```

- El fichero **portal.properties** estará en la raíz del classpath y deberá ser declarado en el **liferay-hook.xml**

```
<hook>
    <portal-properties>portal.properties</portal-properties>
</hook>
```

# Hook de Action de Struts

- Las url de los Action, se compondrán de la siguiente manera

```
http://localhost:8080/c/portal/miejemplo
```

- Las url de los nuevos Action, deberán de cumplir el patrón

```
/portal/
```



# Hook de Action de Struts

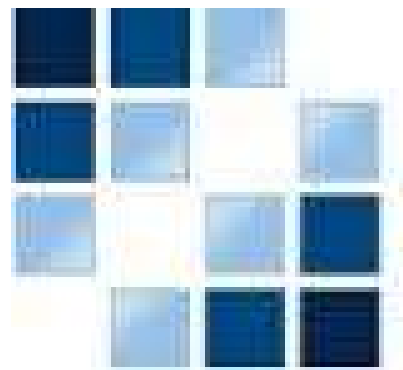
- La nueva clase ha de heredar de **BaseStrutsPortletAction** para sobrescribir Struts ya existentes.
- La herencia de **BaseStrutsPortletAction**, permite implementar el siguiente método

```
public void processAction(StrutsPortletAction originalStrutsPortletAction,  
                          PortletConfig portletConfig, ActionRequest actionRequest,  
                          ActionResponse actionResponse) throws Exception {
```

# Hook de Action de Struts

- La nueva clase ha de heredar de **BaseStrutsAction** para nuevos Action.
- La herencia de **BaseStrutsAction**, permite implementar el siguiente método.

```
@Override  
public String execute(HttpServletRequest request, HttpServletResponse response)  
                                throws Exception {  
    return “/portal/miVista.jsp”;  
}
```



# LIFERAY

Enterprise. Open Source. For Life.

HOOK DE JSP

# Hooks JSP

- Reemplaza una JSP
- Basta con copiar el JSP ya existente en el portal al proyecto, y modificar las características necesarias, indicando en el XML la ruta donde se sitúa en nuevo JSP.
- El editor del XML, permite referenciar a los XML y realiza la copia fácilmente.

# Hooks JSP

- Si únicamente se quiere añadir un trozo de html al jsp existente, no es necesario copiar y pegar todo el código, se puede importar con la etiqueta

```
<liferay-util:include page="/html/portlet/login/login.portal.jsp" />
```

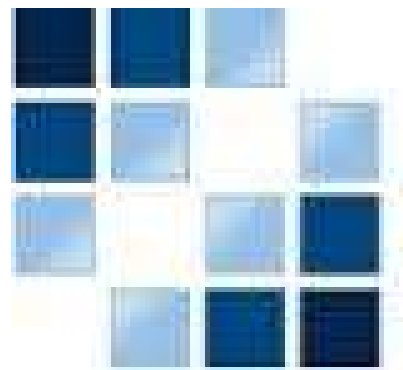
# Hooks JSP

- Se puede manipular la JSP original, para añadir o eliminar alguna parte de forma programática, a partir de la etiqueta **<liferay-util:buffer>**, que crea un String con el JSP de partida.
- Pudiendo realizar sustituciones, eliminaciones a modo de substring.

# Hooks JSP

- Ejemplo de extensión de JSP

```
<liferay-util:buffer var="html">  
    <liferay-util:include page= "/html/portlet/blogs/search.portal.jsp" />  
</liferay-util:buffer>  
  
<% html = StringUtil.add( html, "Didn't find what you were looking for? " +  
    "Refine your search and try again!", "\n");  
%>  
  
<%= html %>
```



# LIFERAY

Enterprise. Open Source. For Life.

## HOOK DE SERVICIO



# Hooks de Servicio

- Toda la funcionalidad de Liferay, esta expuesta por su capa de servicios, los **Hook** de servicios, permiten cambiar el comportamiento de estos servicios, indicando una nueva implementación.
- El procedimiento se basa en extender las clases que proporciona el API, denominadas clases **Wrapper**, en lugar de sobrescribir las clases que representan el servicio directamente.
- Se puede realizar una búsqueda de las clases disponibles

\*ServiceWrapper

# Hooks de Servicio

- Un caso concreto sería el servicio que permite manejar los usuarios del portal.
- Para modificar su funcionalidad, deberíamos extender la clase **UserLocalServiceWrapper** y no la interface **UserLocalService**.
- Entrada en el fichero **liferay-hook.xml**

```
<service>  
    <service-type>com.liferay.portal.service.UserLocalService</service-type>  
    <service-impl>MyUserLocalService</service-impl>  
</service>
```



HOOK DE INDEXADOR DE CONTENIDO

# Hooks del indexador de contenido

- Permite modificar resultados de búsquedas, índices y consultas del portal.
- Por ejemplo, por defecto cuando se añaden usuarios al portal, solo se indexan el nombre y el apellido, pero campos como el título, no, luego búsquedas por el campo título, no serán resueltas.
- Los Hook de indexación, permiten ampliar los campos indexados para cada entidad.

# Hooks del indexador de contenido

- Entrada en el **liferay-hook.xml**

```
<indexer-post-processor>
  <indexer-class-name>
    com.liferay.portal.model.User
  </indexer-class-name>
  <indexer-post-processor-impl>
    com.ejemplo.MiIndexerPostProcesor
  </indexer-post-processor-impl>
</indexer-post-processor>
```

# Hooks del indexador de contenido

- Se ha de definir cual es la entidad sobre la que se va a buscar en el parámetro **indexer-class-name**
- Y la clase que implemente el post-procesamiento de la búsqueda que extienda de **BaseIndexerPostProcessor**, en el parámetro **indexer-post-processor-impl**.

# Hooks del indexador de contenido

- Implementación de **BaseIndexerPostProcessor**

```
@Override
public void postProcessDocument(Document document, Object obj)
                                throws Exception {
    User userEntity = (User) obj;
    String indexerUserTitle = userEntity.getJobTitle();
    if (indexerUserTitle.length() > 0)
        document.addText(Field.TITLE, indexerUserTitle);
}
```