Desarrollo sobre Liferay

Contenidos

1. Introducción
2. Estructura del proyecto
3. Desarrollo de un Tema
4. Thumbnail, ScreenShot y Favicon
5. Estilos
6. Javascript
7. Esquemas de color

INTRODUCCIÓN

Introducción

- Un tema es un plugin que permite modificar el aspecto de las paginas que forman el portal por completo, siendo difícil saber que el portal esta hecho con liferay.
- La forma de crear nuevos temas, se basa en la herencia de otros, modificando solo aquellos aspectos que se quieren personalizar.

Introducción

- La personalización de los Temas, viene marcada por cuatro tipos de recursos
  - CSS
  - Plantillas (Freemarker, Velocity o JSP)
  - Javascript
  - XML
- Se pueden instalar desde el propio Portal accediendo al **Market de Liferay**, o bien creando un Plugin de tipo Tema y desplegandolo.

Plugin de Eclipse

- No hay actualmente mantenidos, la mejor opción es **veloeclipse**.
- Para su instalación en versiones de eclipse posteriores a 3.4, hay que instalar el plugin

Eclipse Test, Example, and Extras / Eclipse 2.0 Style Plugin Support

- Descargable desde

http://download.eclipse.org/eclipse/updates/4.4

- Y luego el propio **veloeclipse**

http://veloeclipse.googlecode.com/svn/trunk/update/

ESTRUCTURA DEL PROYECTO

Estructura del proyecto

- Proyecto **ANT**
  - **/<carpeta de proyecto>/**
    - **/docroot/**
      - **/WEB-INF/**
        - **liferay-look-and-feel.xml**
        - **liferay-plugin-package.properties**
        - **web.xml**
      - **/_diffs/**
        - **/css/**
        - **/images/**
        - **/js/**
        - **/templates/**

Estructura del proyecto

- Proyecto **MAVEN**
  - **/<carpeta de proyecto>/**
    - **/webapp/**
      - **/css/**
      - **/images/**
      - **/js/**
      - **/templates/**
      - **/WEB-INF/**
        - **liferay-look-and-feel.xml**
        - **liferay-plugin-package.properties**
        - **web.xml**

DESARROLLO DE UN TEMA

Desarrollo de un Tema

- Se pueden crear los proyectos de Temas siguiendo **ANT** o **MAVEN**.
- En ambos dos casos, se ha de configurar el Tema padre, que será el Tema de donde se extraigan las características por defecto.
- Una vez definido el proyecto, se empieza la personalización, que se basa en la sobre-escritura de las propiedades que proporciona el padre.

Desarrollo de un Tema

- El Tema padre en los proyectos ANT, se especifica en el **build.xml** del proyecto del **Tema**

```
<?xml version="1.0"?>
<!DOCTYPE project>

<project name="temaPrueba-theme" basedir="." default="deploy">
    <import file="../build-common-theme.xml" />

    <property name="theme.parent" value="_styled" />
</project>
```

Desarrollo de un Tema

- El Tema padre en los proyectos **MAVEN**, se especifica en el **pom.xml** del proyecto del **Tema**

```
<properties>
    <liferay.theme.parent>_styled</liferay.theme.parent>
    <liferay.theme.type>vm</liferay.theme.type>
</properties>
```

Desarrollo de un Tema

- La herencia de los temas se puede hacer solo sobre temas que estén instalados en Liferay.
- Dependiendo del tipo de proyecto creado **MAVEN** o **ANT**, se ha de configurar una propiedad en un fichero o en otro.
  - **MAVEN**
    - En **pom.xml**, dentro del plugin de Liferay, el parámetro de configuración **parentTheme**
  - **ANT**
    - En build.xml, la propiedad **theme.parent**

Desarrollo de un Tema

- Los posibles valores que pueden tomar, por defecto serán
  - **classic**
  - **welcome** (a partir de la versión 6.2)
  - **_styled** (Maquetación mínima)
  - **_unstyled** (Sin maquetación)
- Se pueden emplear otros, siempre que se hayan instalado en Liferay.
- Lo mas habitual es empezar con **_styled**

Desarrollo de un Tema

- Es posible parametrizar los temas, añadiendo los parámetros al fichero **liferay-look-and-feel.xml** con la etiqueta

```
<settings>
        <setting key=”mi-setting” value=”mi-valor-1” />
<settings>
```

- Siendo estos parámetros accesibles desde las plantillas de **Velocity** con

```
$theme_display.getThemeSetting("mi-setting")
```

Desarrollo de un Tema

- La etiqueta **setting** tiene los siguientes parámetros
  - Key: Clave para acceder a la propiedad
  - Configurable: Booleano que permite editar la propiedad.
  - Type: Tipo de control para representar el parametro
    - Select
    - checkbox
  - Options: Valores posibles para el tipo select.
  - Value: Valor por defecto para el parametro.

Desarrollo de un Tema

- Los temas, tienen una serie de plantillas creadas para cada uno de los tipos de recursos que se pueden encontrar en el sitio, estas plantillas están en la carpeta

```
webapp/templates
```

- Las hay de
  - **Freemarker**, extensión **ftl**
  - **Velocity**, extensión **vm**
- La posibilidad de desarrollar en JSP, no es recomendada, por presentar problemas de rendimiento en la generación dinámica de recursos.

Desarrollo de un Tema

- Centrándose en **Velocity**, se encuentran las siguientes plantillas
  - **init_custom.vm**: Fichero para la definición de variables a emplear en el resto de plantillas
  - **init.vm**: Fichero con las variables que define Liferay, que son empleadas en el resto de plantillas.
  - **navigation.vm**: Menú principal de navegación.
  - **portal_normal.vm**: Estructura general HTML de todas las paginas del portal.

Desarrollo de un Tema

- Centrándose en **Velocity**, se encuentran las siguientes plantillas (**Continuación**)
  - **portal_pop_up.vm**: Estructura HTML de los popups que nos encontramos dentro del portal.
  - **portlet.vm**: Estructura HTML de cada uno de los portlets. Básicamente es un section con su encabezado, el menú de opciones de cada portlet y el contenido del mismo.

Desarrollo de un Tema

- Se puede acceder a los Servicios de Liferay desde las plantillas de Velocity

```
#set($layoutLocalService = $serviceLocator
                        .findService("com.liferay.portal.service.LayoutLocalService"))

#set($xyz = $layoutLocalService
                        .getFriendlyURLLayout($layout.getOwnerId(), "/home"))
```

- En el caso de que el servicio pueda retornar una excepción

```
#set($userLocalService = $serviceLocator
                .findExceptionSafeService("com.liferay.portal.service.UserLocalService"))

#set($user = $userLocalService.getUserById($getterUtil.getLong("12345"))
```

Desarrollo de un Tema

- Y así con el resto de recursos, una vez esta todo el tema, se despliega, para ello
  - En **ANT** se lanza la tarea
    - Ant clean deploy
  - En **MAVEN**
    - mvn clean
    - mvn liferay:theme-merge
    - mvn liferay:deploy

Desarrollo de un Tema

- En **ANT**, se generan los recursos dentro de la carpeta **_diffs**, y al desplegar se mezclan con lo que haya en el raíz, que será lo obtenido por herencia.
- En **MAVEN**, no son visibles los ficheros del padre, hasta que no se mezcla, el proyecto tiene los ficheros personalizados, el resultado final se genera en la carpeta **target**.

Thumbnail, ScreenShot y Favicon

- Sobre-escritura de Thumbnail, Screenshot y Favicon del tema.
  - Screenshot, se recomienda un ancho de 1024px.
  - Thumbnail, se recomienda un ancho de 150px.
- Si estamos con un proyecto **ANT**, los ficheros se copian dentro de una carpeta

```
docroot/_diffs/images
```

- Si estamos en uno **MAVEN**, en la carpeta

```
webapp/images
```

Estilos

- Para modificar los estilos, se dispone de dos ficheros css.
  - **main.css**: Importación de todos los estilos, se podría añadir uno nuevo
  - **custom.css**: Fichero vacío, que se importa en ultimo lugar, destinado a incluir los estilos personalizados.

Uso de Javascript

- En los Temas, existe un único fichero javascript, **main.js**, que se ejecuta siempre que se carga una página.
- Si se desea añadir un nuevo js al Tema, este se ha de añadir al fichero **portal-normal.vm**, como se haría en cualquier HTML, empleando la variable velocity **$javascript_folder**.

```
<script src="$javascript_folder/miFichero.js" type="text/javascript"/>
```

- Liferay desde la versión 6.0, viene con un framework javascript **Alloy UI**.

Uso de Javascript

- En el fichero **main.js**, se encuentran definidos 3 funciones javascript, donde se puede incluir código
  - **AUI().ready**: Se ejecuta cuando se termina de carga la pagina
  - **Liferay.Portlet.ready**: Se ejecuta cada vez que se cargue un portlet de la página actual.
  - **Liferay.on**: Se ejecutará después de que todos los portlets de la página estén cargados.

Uso de Javascript

- Por defecto liferay minimiza los ficheros js y css para que sea mas rápido su carga, esto se puede cambiar con propiedades del **portal.properties**
  - javascript.fast.load=false
  - theme.css.fast.load=false
  - minifier.enabled=false

Esquemas de color y recursos gráficos

- Un esquema de color es una variante de un Tema, en el que se mantienen el Tema original únicamente variando la gama de colores empleados.
- Para definirlo basta con crear una nueva entrada en el fichero "**WEB-INF/liferay-look-and-feel.xml**", dentro de la etiqueta "**theme**", que sea "**color-scheme**".

Esquemas de color y recursos gráficos

- En ese nodo, se hace referencia a aquellos recursos que hayan sido modificados, que podrán ser
  - **CSS**
  - **Imágenes**

Esquemas de color y recursos gráficos

- Para ello, se tiene los sub-nodos
  - **Default-cs.** Para indicar si es el por defecto (true/false).
  - **css-class**. Nombre de la clase CSS que se añadirá al nodo principal para cambiar estilos y adecuarlos al esquema de color.
  - **color-scheme-images-path**. Para esta propiedad, es interesante el empleo de la variable **${images-path}**. Suele emplearse valores como
    - ${images-path}/color_schemes/Defecto

Esquemas de color y recursos gráficos

- Por ultimo se han de cargar los CSS desde **main.css**
  - @import url(color_schemes/Defecto.css);