

Lab 5 – Utilización de templates

Objetivos del taller:

Personalizar las aplicaciones introducción cambios en los templates empleados por el generador de JHS.

Paso 0. Adicionar los copyright en todas las páginas de la aplicación.

DETERMINAR EL SEGMENTO DE CODIGO.

- Edite una de las páginas generadas. Por ejemplo Countries.jspx.
- Adicione el facet appCopyright.
- Adicione a este facet un componente de tipo OutputText.
- Observe el código jsp, según la modificación aplicada.

```

225
226 <!-- DEBUG:END:FORM_PAGE_CONTENT : default/page/formPageContent.vm, nesting level:
227
228 <f:facet name="appCopyright">
229     <h:outputText value="Propiedad intelectual de Seguros Bolívar"/>
230 </f:facet>
231
232 </af:panelPage>
233 </af:form>
234 </afh:body>
235 </afh:html>
236 </f:view>
237 </jsp:root>
238
  
```

- Deducir cual es el template que hay que modificar para hacer esta modificación permanente.

CREAR UN NUEVO TEMPLATE

- Crear el nuevo subdirectorio “\ViewController\templates\asw\page” para almacenar los templates personalizados y copiar en el template dataPage.vm.
- Editar el nuevo archivo dataPage.vm y modificarlo introducir el facet appCopyright:

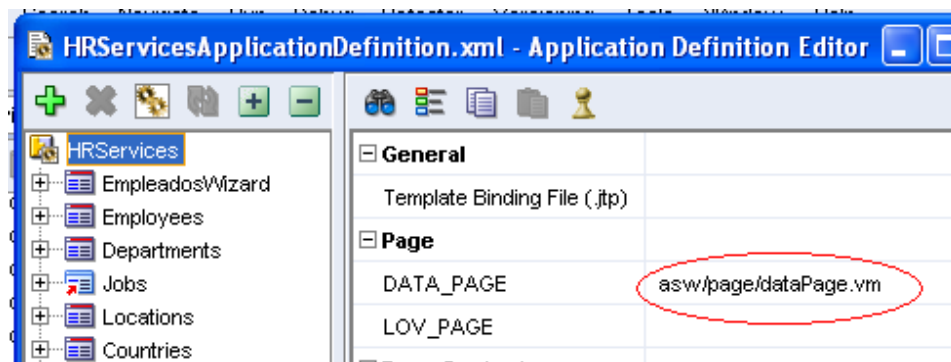
```

dataPage.vm
52      #JHS_PARSE("PAGE_MENU" ${JHS.current.model})
53      <f:facet name="messages">
54          <af:messages id="messages"/>
55      </f:facet>
56
57      #JHS_PARSE(${JHS.page.contentTemplateIdentifier} ${JHS.current.model})
58
59      <f:facet name="appCopyright">
60          <h:outputText value="Propiedad intelectual de Seguros Bolivar"/>
61      </f:facet>
62
63      </af:panelPage>
64  </af:form>
65  </afh:body>
66  </afh:html>
67 </f:view>
68 </jsp:root>

```

EMPLEAR EN LA GENERACION UN TEMPLATE DIFERENTE

- Editar el application definition para especificarle al generador el nuevo template que se debe emplear para generar el esqueleto de las páginas JSF-JSP. Esta especificación se hace a nivel de Service, garantizando que el cambio va a repercutir en todas las páginas.



Paso 1. Cambiar la imagen corporativa de la aplicación.

CREAR UNA NUEVA REGION

- Determine la imagen que desee emplear como logo de la aplicación y almacénelo en el directorio ViewController/public_html/images/

- Crear una nueva región (archivo jsp que será empleado como contenido de un facet). Realice una copia del archivo ViewController/public_html/common/regions/branding.jspx bajo el nombre aswBranding.jspx
- Edite el archivo aswBranding.jspx y modifique la referencia a la imagen seleccionada.

DECLARAR LA NUEVA REGION

- Edite el archivo WEB-INF/regions-metadata.xml, e introduzca la definición de un nuevo componente:

```
<component>
  <component-type>asw.cursos.jhs.branding</component-type>
  <component-class>oracle.adf.view.faces.component.UIXRegion</component-class>
  <component-extension>
    <region-jsp-ui-def>/common/regions/aswBranding.jspx</region-jsp-ui-def>
  </component-extension>
</component>
```

MODIFICAR EL TEMPLATE

- Editar el nuevo archivo dataPage.vm y modificarlo para que el facet branding referencie el componente-type asw.cursos.jhs.branding:

```
<f:facet name="branding">
  <af:region id="branding" value="#{bindings}"
    <del>regionType="#{bindings}"</del>
    regionType="asw.cursos.jhs.branding"/>
</f:facet>
```

- Qué debe hacer para que sólo aparezca la imagen de BRANDING?

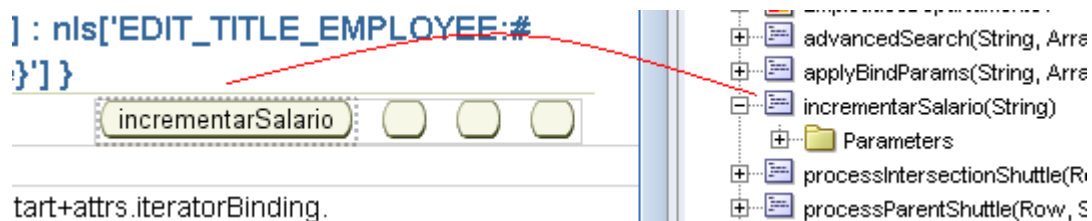
Paso 2. Adicionar a la forma de empleados el botón “incrementar salario”. Este botón invoca el método incrementarSalario del application module.

- Crear el método incrementarSalario en el application module HRServices, y publicarlo para que puede ser invocado desde el cliente

```
public void incrementarSalario(String employeeId) {
    EmployeesViewImpl vo = this.getEmployeesView1();
    Key llave = new Key(new Object[] {employeeId});

    Row[] empleados = vo.findByKey(llave, 1);
    if (empleados != null && empleados.length>0) {
        EmployeesViewRowImpl emp = (EmployeesViewRowImpl)empleados[0];
        Number nuevoSalario = emp.getSalary().multiply(1.1);
        emp.setSalary(nuevoSalario);
    }
}
```

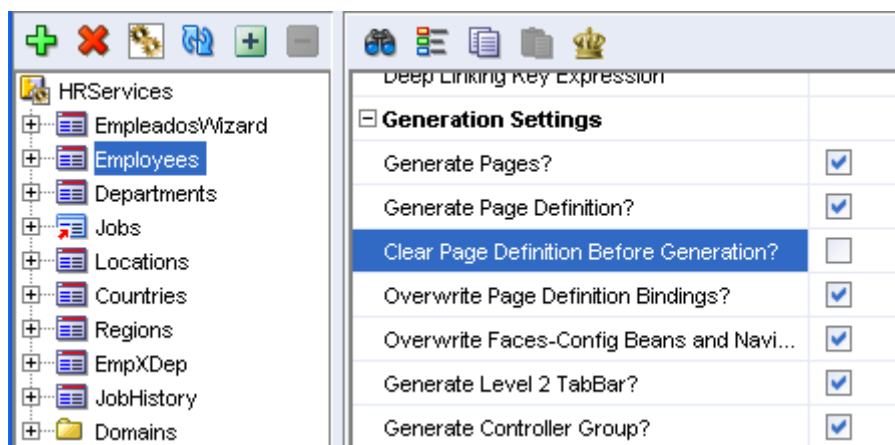
- Editar la página Employees.jspx y adicionar el botón de incrementar desde JDeveloper. Asegurarse que el parámetro enviado al método sea el identificador del empleado (bindings.EmployeesEmployeeId.inputValue).



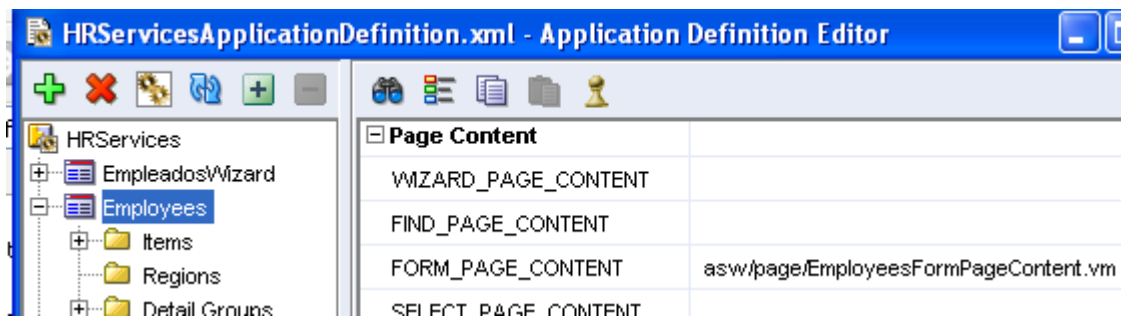
- Correr la aplicación modificada y comprobar el funcionamiento del botón.

Paso 3. En el paso anterior se introdujeron cambios post-generación que se perderán si se regenera la aplicación. Para mantener estos cambios es necesario :

- Evitar que las definiciones aplicadas en el Page Definition de la página se pierdan



- Determinar el template que es necesario personalizar. Editar la página Employees.jspx y observar los comentarios del generador. En este caso se decide hacer el cambio en el template formPageContent.vm.
- Hacer una copia de formPageContent.vm a asw/page/EmployeesFormPageContent.vm.
- Introduzca en el archivo EmployeesFormPageContent.vm la etiqueta “<af:commandButton>” correspondiente al botón “incrementar salario”.
- Especificar en el Application Definition que el grupo Employees ahora emplea el template EmployeesFormPageContent.vm.



Paso 4. Internacionalización. Con el paso anterior la etiqueta del botón incrementar quedó “quemada” en el template, por lo cual no podrá ser trasladada a otro idioma sin regenerar la aplicación.

- Emplear la expresión #{nls[Key]} como label del botón:

```
<af:commandButton actionListener="#{bindings.incrementarSalario.execute}"
    text="#{nls['LABEL_INCREMENTAR_SALARIO']}"
    disabled="#{!bindings.incrementarSalario.enabled}"/>
```

- Insertar la entrada LABEL_INCREMENTAR_SALARIO manualmente en el ApplicationResources.properties.

Sin embargo, la solución óptima es emplear la función JHS.nls, que hace los dos pasos anteriores automáticamente:

```
<af:commandButton actionListener="#{bindings.incrementarSalario.execute}"
    text="#{JHS.nls('Incrementar salario', 'LABEL_INCREMENTAR_SALARIO')}"
    disabled="#{!bindings.incrementarSalario.enabled}"/>
```

Paso 5. Crear hiperenlaces entre grupos. Crear un hiperenlace desde la página de DEPARTMENTS que permita ir a editar al MANAGER. (El manager del departamento se edita en la página de EMPLOYEES).

CREAR EL DEEP LINKING HACIA EL GRUPO EMPLOYEES

- Abrir el Application Definition Editor y seleccionar el grupo Employees y modificar las propiedades Deep Linking

<div> <div>nrServices</div> <div> <div>Empleados\Wizard</div> <div>Employees</div> <div>Items</div> <div>Regions</div> <div>Detail Groups</div> <div>Departments</div> </div> </div>	<div> <div>Deep Linking</div> <div> <div>Type of Deep Linking</div> <div>Query By Key Value</div> </div> <div> <div>Enable Deep Linking Expression</div> <div>#{jsfNavigationOutcome=="DeepLink\$GROUP_NAME\$"} </div> <div> <div>Deep Linking Key Expression</div> <div>#{param.managerId} </div> </div> </div> <div> <div>Generation Settings</div> </div> </div>
--	---

Con esto se está especificando: “Se puede llegar a la página Employees por medio del outcome (reglas de navegación) DeepLinkEmployees, recibiendo como parámetro el valor param.managerId. El valor de este parámetro será empleado como filtro por la llave primaria”.

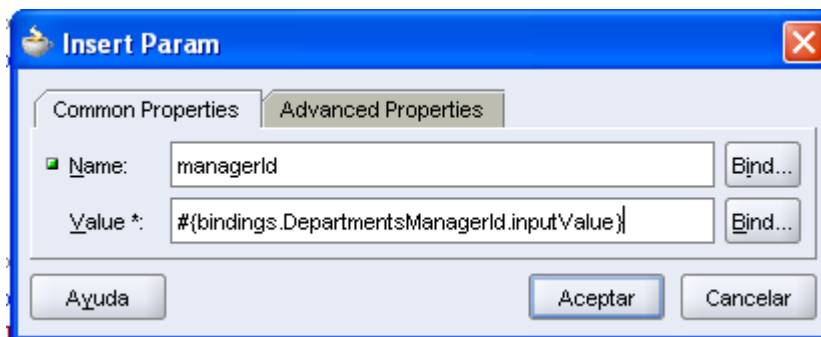
- Regenere la aplicación para que se actualicen las reglas de navegación.

CREAR EL HIPERENLACE EN DEPARTMENTS.JSPX

- Verificar que el item ManagerId tenga Display Type TextInput. (En caso de ser necesario cámbielo y regenere).
- Editar la página Departments.jspx y cambiar (convert) el UI Component del campo ManagerID a CommandLink. Cambie el text CommandLink a “#{bindings.DepartmentsManagerId.inputValue}” y asignar al Action el outcome DeepLinkEmployees.

<div> <div>General</div> </div>	
<div> <div>Text</div> </div>	#{bindings.DepartmentsManagerId.inputValue}
<div> <div>Action</div> </div>	DeepLinkEmployees
<div> <div>ActionListener</div> </div>	

- Revisar el código fuente y eliminar código basura que haya quedado como parte del CommandLink.
- Inserte un componente tipo “param” como hijo del CommandLink.



The dialog box titled "Insert Param" has two tabs: "Common Properties" and "Advanced Properties". Under "Common Properties", there are two fields: "Name" with the value "managerId" and "Value" with the value "#{bindings.DepartmentsManagerId.inputValue}". Each field has a "Bind..." button to its right. At the bottom of the dialog are three buttons: "Ayuda", "Aceptar", and "Cancelar".

- Pruebe la aplicación.