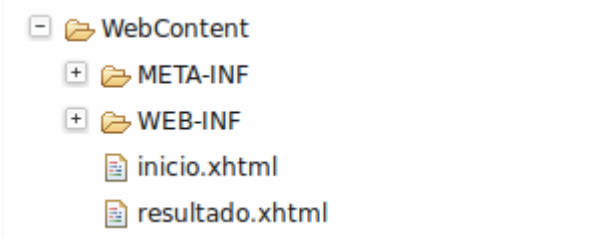
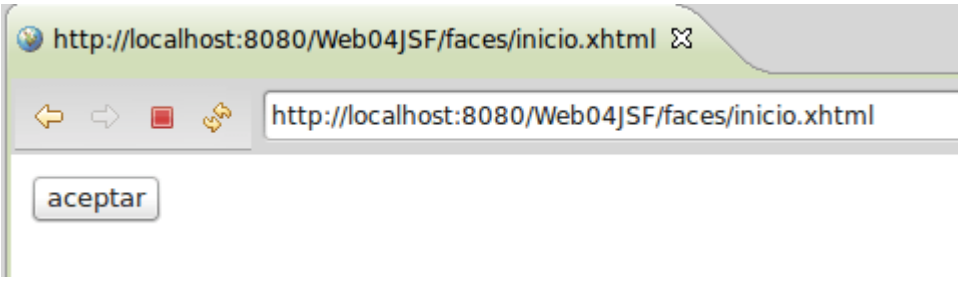
En los últimos capítulos del libro hemos trabajado con JSF como framework de capa de presentación y hemos construido varias páginas entre las que navegamos .Sin embargo la navegación que el libro define es una navegación estática ya que esta completamente ligada a la estructura de las páginas. A continuación se muestran dos páginas ligadas de forma estática.



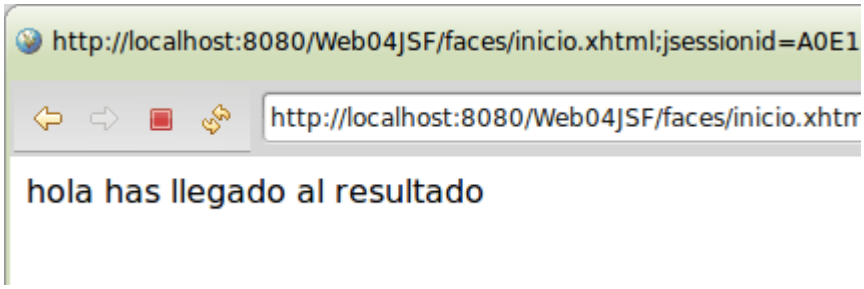
La página de inicio contiene el siguiente formulario JSF con un único boton:

<h:form>
<h:commandButton id=”boton” value=”aceptar” action=”resultado”/>
</h:form>

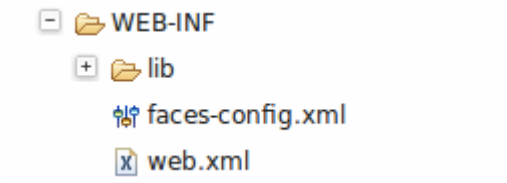
Al solicitar la página al servidor nos mostrará la siguiente información.



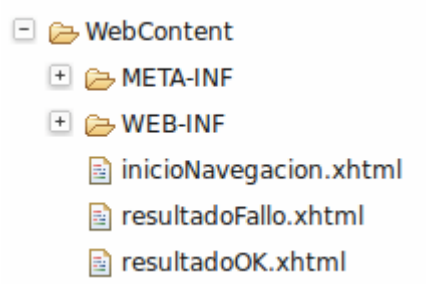
Al pulsar el boton el framework de JSF nos enviara a la pagina xhtml que este definida en el atributo action (action=”resultado”) en este caso la página de resultado.xhtml que nos imprimirá el siguiente mensaje.



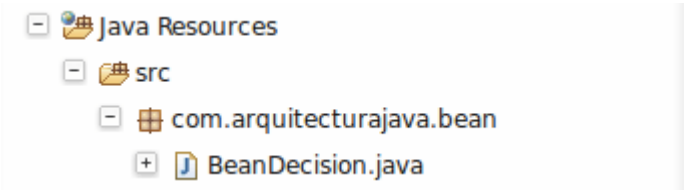
Aunque el funcionamiento es correcto según la aplicación aumente de tamaño necesitaremos una forma de navegar entre paginas JSF mas flexible .Para ello usaremos el fichero faces-config.xml y sus capacidades a la hora de definir reglas de navegación. Recordemos que este fichero se encuentra ubicado en el directorio WEB-INF.



Una vez que hemos definido este fichero .Vamos a utilizar la navegación dinamica apoyandonos en las siguientes 3 páginas.



La página de inicioNavegación.xhtml es nuestro punto de partida y para decidir que página de destino carga se apoyará en un ManagedBean definido por la aplicación. En nuestro caso hemos decidido denominar a este ManagedBean “BeanDecision”



El código de este ManagedBean es realmente sencillo:

```
package com.arquitecturajava.bean;

import javax.faces.bean.ManagedBean;

@ManagedBean
public class BeanDecision {

private String nombre;

public String getNombre() {
 return nombre;
 }

public void setNombre(String nombre) {
 this.nombre = nombre;
 }

public String decide() {

if (nombre.equals("cecilio")) {

return "exito";
```

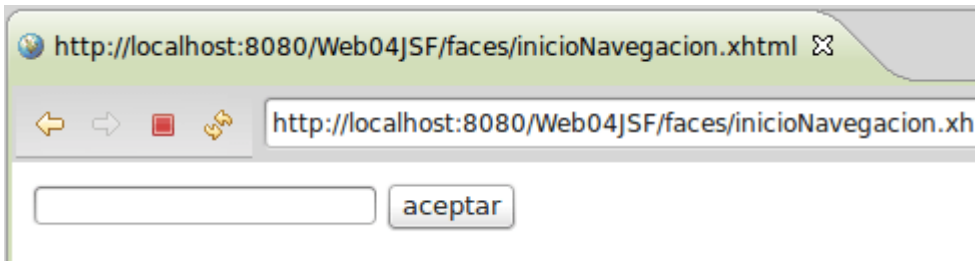
```
} else {

return "fallo";

}

}
}
```

Una vez hemos definido el ManagedBean podemos solicitar la página “inicioNavegacion.xhtml” la cual incluye una caja de texto que podemos rellenar.



Si revisamos el código fuente de esta página podremos ver como el botón aceptar del formulario esta ligado a un método concreto del ManagedBean

<h:form>
<h:inputText value=”#{beanDecision.nombre}”/>
<h:commandButton id=”boton” value=”aceptar” action=”#{beanDecision.decide}”/>
</h:form>

El método “decide” nos devuelve “exito” o “fallo” dependiendo del valor de la cadena que este ligada al atributo “nombre” del ManagedBean. Ahora bien esto no nos solventa del todo la navegación entre las distintas páginas . Para que el framework JSF pueda dirigirmos a las páginas que nosotros queremos .deberemos definir unas reglas de navegación básicas a

nivel de faces-config.xml.

```
<navigation-rule>
<from-view-id>/inicioNavegacion.xhtml</from-view-id>
<navigation-case>
<from-outcome>exito</from-outcome>
<to-view-id>/resultadoOK.xhtml</to-view-id>
</navigation-case>
<navigation-case>
<from-outcome>fallo</from-outcome>
<to-view-id>/resultadoFallo.xhtml</to-view-id>
</navigation-case>
</navigation-rule>
```

En este caso la regla de navegación es muy sencilla por un lado definimos cual es la página de partida.

```
<from-view-id>/inicioNavegacion.xhtml</from-view-id>
```

Por otro lado definimos el outcome o resultado que esperamos.

```
<from-outcome>exito</from-outcome>
```

Una vez el outcome esta claro simplemente definimos la vista destino a la que deseamos ir basandomos en ese outcome

```
<to-view-id>/resultadoOK.xhtml</to-view-id>.
```

A partir de ahora podremos eliminar la navegación estática de las páginas y centrarnos en una navegación dinámica mucho mas flexible.