AUT06_02 Introducción a JSF 2

1. Ejercicio – Añadir inserción de actores

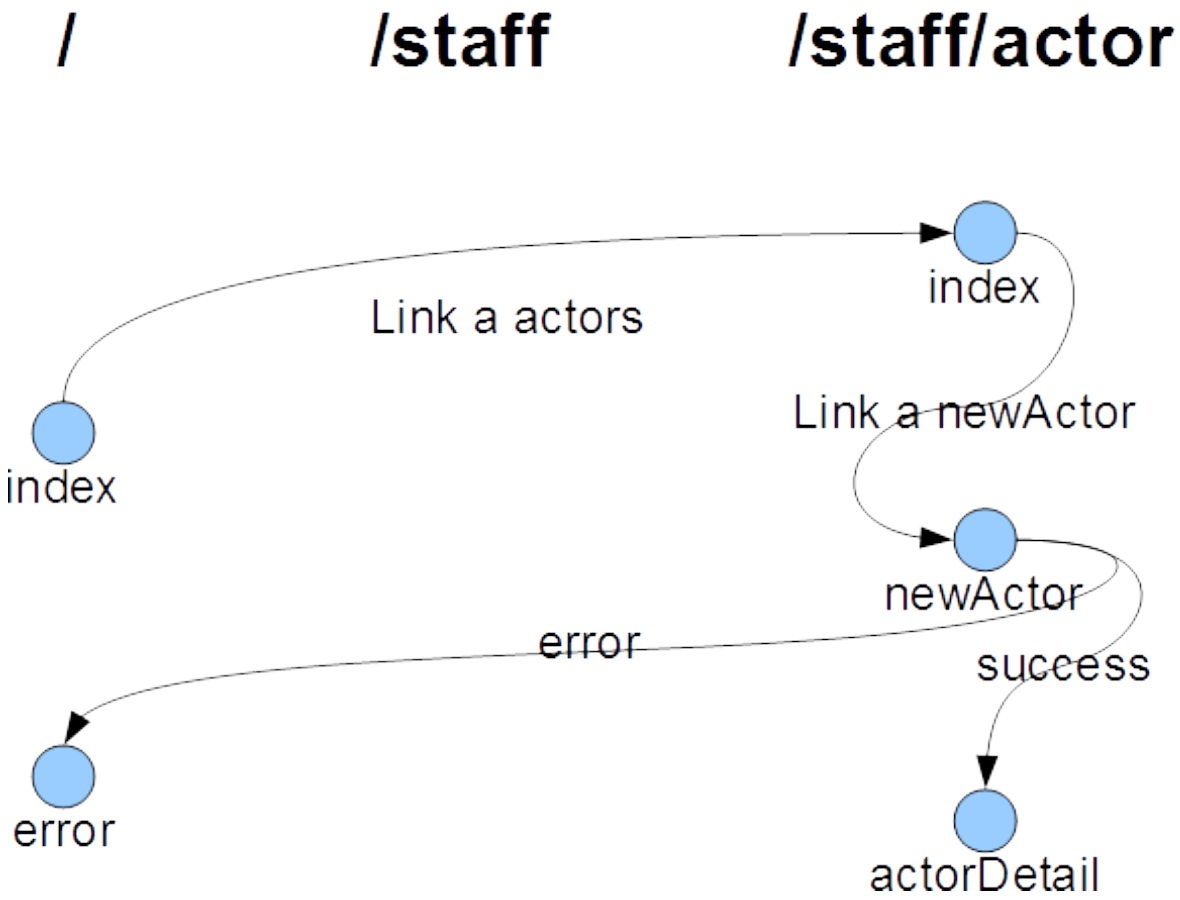
1.1 Navegación

i. Desde la página index habrá un link a /staff/actor/index, donde se mostrará la lista de actores y un link a **newActor.xhtml**

ii. En newActor habrá un formulario que invocará a **actorBean.save()**

   i. Si todo va bien, el método save devolverá "actorDetail", navegando a la página de detalle de actor

   ii. En caso de error se devolverá "/error" yendo a la página de errores general (en /)



1.2 Objetos DAO, ValueObject/DTO

La aplicación hará uso de las clases del patrón DAO creadas en el ejercicio anterior.

El DAO tiene que incorporar las siguientes operaciones, para MySQL:

1. Insertar un actor, devolviendo el método insert el POJO del actor insertado, **incluyendo el actorId generado**. El nuevo método tendrá la siguiente signatura (nombre): **+insert(element: Actor): Actor**

Para obtener el actorId generado hay que hacer lo siguiente:

1. En el constructor de PreparedStatement, añadir un segundo argumento con valor *Statement.RETURN_GENERATED_KEYS*
2. Justo a continuación de la llamada a *executeUpdate* del PreparedStatement, hay que llamar a un método *getGeneratedKeys* que devuelve un ResultSet con el valor generado en la columna 1, de la primera fila. Utilizar el getXXX apropiado (Short, Int, etc.)

1.3 Cambio en página /staff/actors/index

Se añade, encima de la tabla, un link a *nuevo actor*:

```
<p><h:link outcome="newActor" value="Nuevo actor"/></p>
```

1.4 Nueva página newActor (en /staff/actors/newActor)

La página mostrará un formulario para añadir un nuevo actor, así como elementos message para mostrar los errores.

Contendrá un commandButton para invocar el save de actorBean.

```
<h:form>
    <p:panel id="pnlNewActor" header="Nuevo Actor">
        <p:messages id="msgs"/>

        <h:panelGrid id="pnlActor" columns="3" cellpadding="5">
            <f:facet name="header">
                <h:outputText value="Nuevo actor"/>
            </f:facet>
            <h:outputLabel for="txtFirstName" value="Nombre"/>
            <p:inputText id="txtFirstName" value="${actorBean.firstName}"/>
            <p:message for="txtFirstName" display="icon" />
            <h:outputLabel for="txtLastName" value="Apellidos"/>
            <p:inputText id="txtLastName" value="${actorBean.lastName}"/>
            <p:message for="txtLastName" display="icon" />
               <p:commandButton value="Save" update="pnlNewActor" action="#{actorBean.save}"
icon="ui-icon-check" />
        </h:panelGrid>
    </p:panel>
</h:form>
```

1.5 Nueva página actorDetail (en /staff/actors/actorDetail)

Deberá mostrar los datos del actor insertado. (parecido a la página anterior).

Utilizad lo aprendido en la anterior para generar esta nueva, también se pueden mirar ejemplos en:

http://www.primefaces.org/showcase/index.xhtml

1.6 Nueva página error (en /)

Deberá mostrar los mensajes de error.

Ver: http://www.primefaces.org/showcase/ui/message/messages.xhtml

1.7 Cambios en ManagedBean

1.7.1 Bean Validation

Vamos a añadir anotaciones de Bean Validation (JSR 303), sobre los campos **firstName** y **lastName**, de forma que no sean nulos.

Las anotaciones no vamos a definirlas sobre el POJO Actor, lo que vamos a hacer es redefinir los métodos getter de firstName y lastName, en ActorBean (que decíamos hereda de Actor) y justo encima de los getter redefinidos añadiremos la restricción @NotNull. Ejemplo:

```
@Named(value = "actorBean")
@RequestScoped
public class ActorBean extends Actor {

    @NotNull(message = "Apellidos no puede estar vacío")
    @Override
    public String getLastName() {
        return super.getLastName();
    }
...
```

1.7.2 Método save

Vamos a añadir un método **+save(element: Actor): String** al ActorBean. El método llamará al insert del DAO, pasando el propio objeto ActorBean (que hereda de Actor y es el backing bean del formulario anterior).

Si todo va bien, se devuelve “index” indicando que volvemos a la página index que muestra la lista de actores, en caso contrario devolvemos “/error” añadiendo un mensaje de error.

Para añadir un mensaje de error:

```
FacesContext.getCurrentInstance().addMessage(null,
                        new FacesMessage(FacesMessage.SEVERITY_ERROR, "Error", "Error de
inserción"));
```