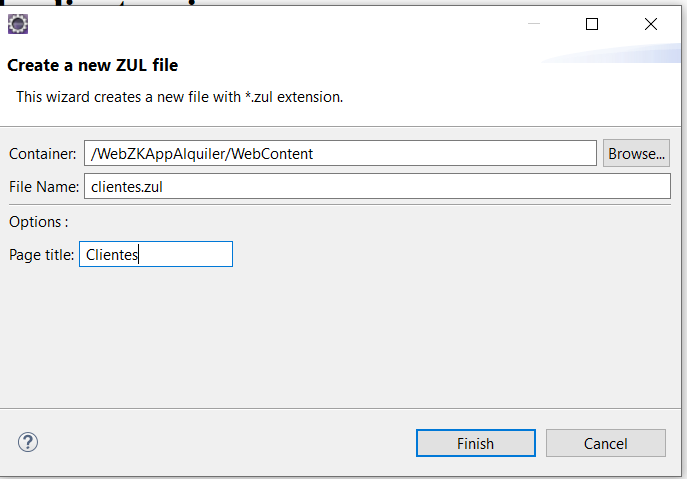
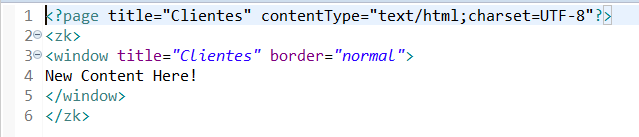
Descomprimir el proyecto ZKAppWeb y abrir en eclipse

Vamos a crear una interfaz que permita visualizar un listado de los clientes, así como poder agregar

Agregamos un nuevo archivo zul



En este nuevo archivo vamos a trabajar la ventana



En la ventana vamos a definir dos áreas

Una para buscar y agregar, y otra para mostrar los clientes

El área de búsqueda va a agrupar los elementos en un Hbox. Este contenedor ordena horizontalmente los componentes que contenga

<hbox align="center">

<hbox align="left">

Texto:

<textbox id="txtBuscar" />

<button id="btnBuscar" label="Buscar"/>

</hbox>

<hbox align="right">

<button id="btnNuevo" label="Nuevo Cliente"/>

</hbox>

</hbox>

El área para mostrar el listado de los clientes será un listbox

<listbox id="clienteListBox" height="160px" emptyMessage="No se encontraron clientes">

<listhead>

<listheader label="Cedula" width="20%"/>

<listheader label="Nombre" />

<listheader label="Dirección" />

<listheader label="Email" />

</listhead>

<listitem>

<listcell label="cedula"></listcell>

<listcell label="nombre"></listcell>

<listcell label="direccion"></listcell>

<listcell label="email"></listcell>

</listitem>

</listbox>

Guardar los cambios y probar el zul

**Modelo Vista Controlador**

Se va a aplicar el patrón de diseño, Modelo Vista Controlador (MVC).

El modelo está ya desarrollado con la clase Cliente y el dao correspondiente

Vamos a crear un controlador para manejar la interacción del usuario con la interfaz.

Para crear un controlador hay que extender la clase org.zkoss.zk.SelectorComposer:

Crearemos un nuevo paquete app.controlador y en este paquete, una nueva clase ClientesController

public class ClientesController extends SelectorComposer<Component> {

}

Después de crear el controlador, lo asociamos al componente de la vista correspondiente. Asociar un controlador con un componente requiere que indiquemos el nombre completo de la clase en el atributo «**apply**» en la página zul.

<window title="Clientes" width="600px" border="normal" apply="app.controlador.ClientesController">

Vamos a cambiar el listBox para mostrar los clientes.

<listbox **model="${$composer.clientes}"** id="clienteListBox" height="160px" emptyMessage="No se encontraron clientes">

<listhead>

<listheader label="Cedula" width="20%"/>

<listheader label="Nombre" />

<listheader label="Dirección" />

<listheader label="Email" />

</listhead>

**<template name="model">**

**<listitem>**

**<listcell label="${each.cedula}"></listcell>**

**<listcell label="${each.nombre}"></listcell>**

**<listcell label="${each.direccion}"></listcell>**

**<listcell label="${each.email}"></listcell>**

**</listitem>**

**</template>**

</listbox>

El texto marcado en negritas, es el código que ha cambiado. Ahora el listbox va a obtener los datos de la variable clientes

La información de cada cliente estará en un template que lleva de nombre model

En el archivo controlador hay que proveer la variable clientes, lo haremos por medio de la notación WireVariable

@WireVariable

private List<Cliente> clientes;

Para esta variable se provee el método get, que será el que se encargue de recuperar los datos a través del ejb

private EJBClienteLocal clienteServ = new EJBCliente();

public ListModel<Cliente> getClientes() {

EJBClienteLocal clienteServ = new EJBCliente();

clientes = (List)clienteServ.findAll();

return new ListModelList<Cliente>(clientes);

}

También en el controlador se definen como atributos aquellos componentes de la interfaz de los que escuchamos sus eventos mediante la anotación **@Wire**.

@Wire

private Textbox txtBuscar;

@Wire

private Listbox clienteListBox;

Luego de estos cambios probamos el archivo clientes.zul

