



Construyendo una aplicación con ZK

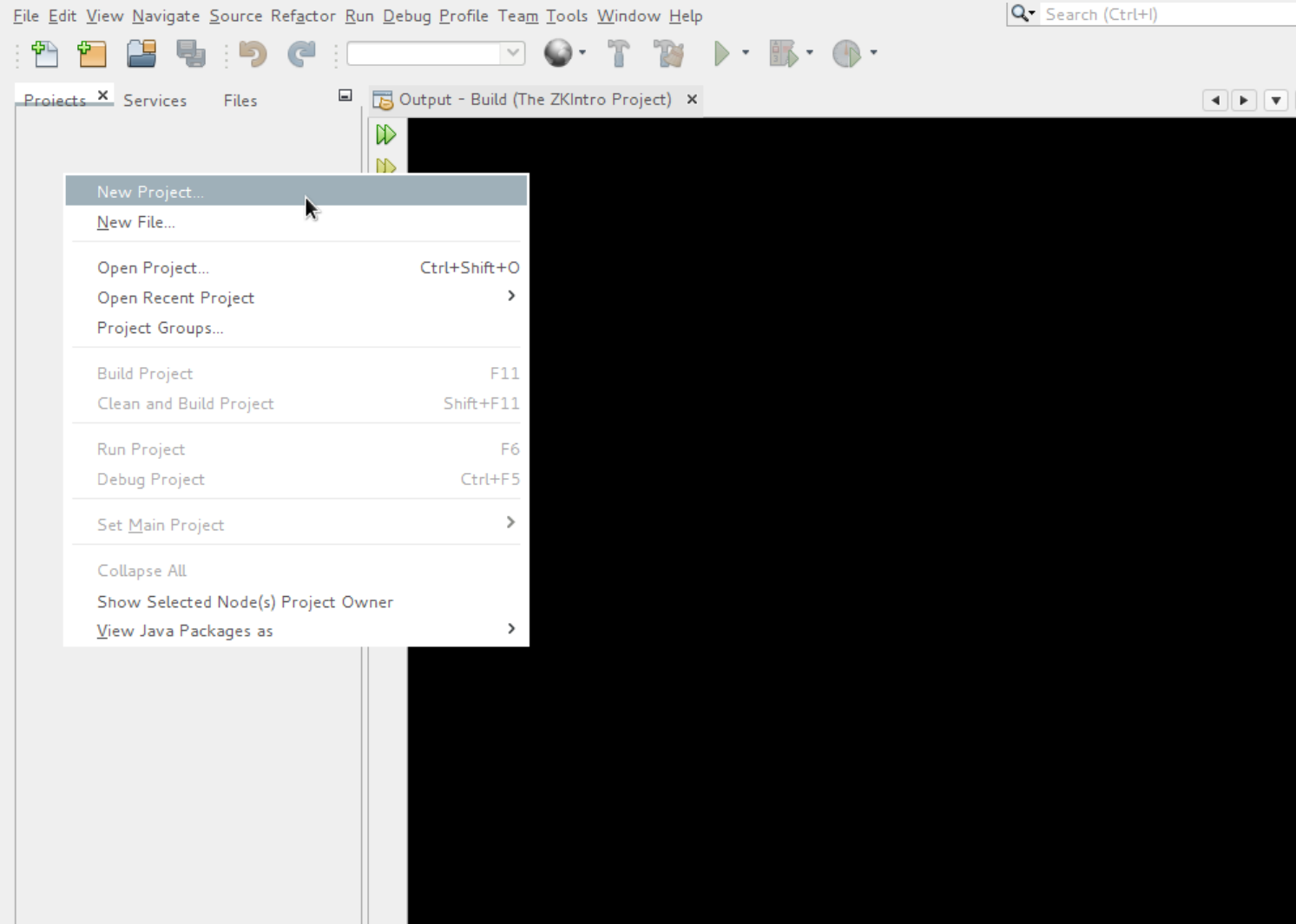
¿Qué es ZK?

- A Java framework for building rich Ajax and mobile applications
 - <http://www.zkoss.org/product/zk>

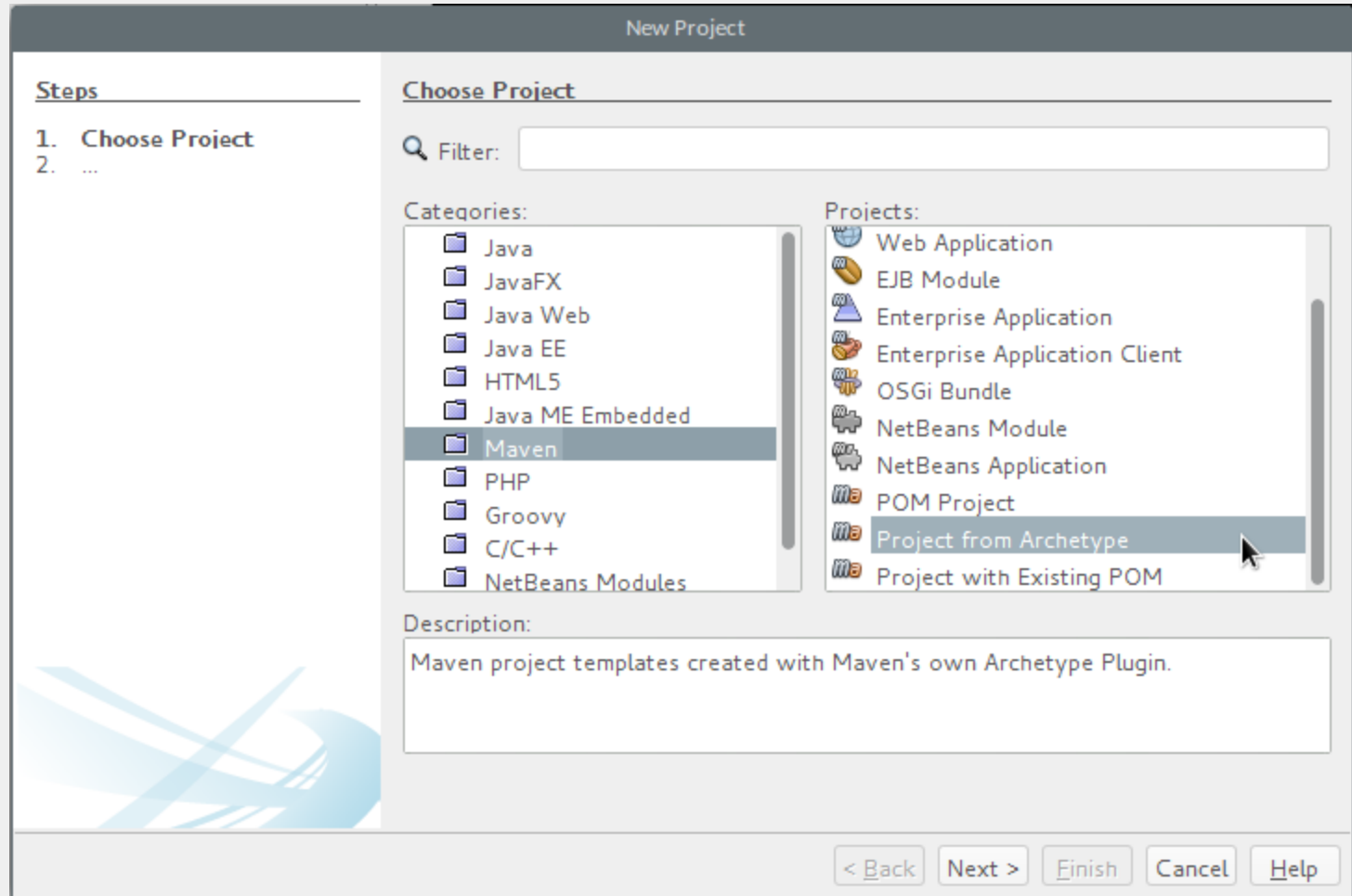
Creando el proyecto de ejemplo

- Maven al rescate
 - Utilizar el arquetipo zk-archetype-webapp
- Utilizaremos una base de ejemplo en PostgreSQL
 - Pagila : base de datos de ejemplo de un video club
- Adicionalmente utilizaremos JOOQ y C3P0 para acceder a la base de datos y mapear las tablas contra objetos en Java
- Jetty como servidor web (embebido)

Crear un nuevo proyecto Maven (NetBeans 8.0.2)



Seleccionar Maven -> Projects from Archetype



Indicar el arquetipo

New Project

Steps

1. Choose Project
2. **Maven Archetype**
3. Name and Location

Maven Archetype

Search: ☐ Show Older

Known Archetypes:

- javate-zk-archetype
- javate-zk-archetype
- javate-zk-jpa-archetype
- kotlin-simple-archetype
- zk-archetype-component
- zk-archetype-extension
- zk-archetype-theme
- zk-archetype-webapp**
- zk-ee-eval-archetype-webapp
- zk-ee-eval-archetype-webapp-spring
- zk-ee-eval-archetype-webapp-spring-jpa

Group ID: Artifact ID:

Version: Repository:

Description:

An archetype that generates a starter ZK CE webapp project

< Back Next > Finish Cancel Help

Darle un nombre al proyecto

New Project

Steps

1. Choose Project
2. Maven Archetype
3. **Name and Location**

Name and Location

Project Name: VideoClub

Project Location: /home/mrojas/Development/Projects/NetBeansProjects [Browse...](#)

Project Folder: /home/mrojas/Development/Projects/NetBeansProjects/VideoClub

Artifact Id: VideoClub

Group Id: mx.ihsa.videoclub

Version: 1.0-SNAPSHOT

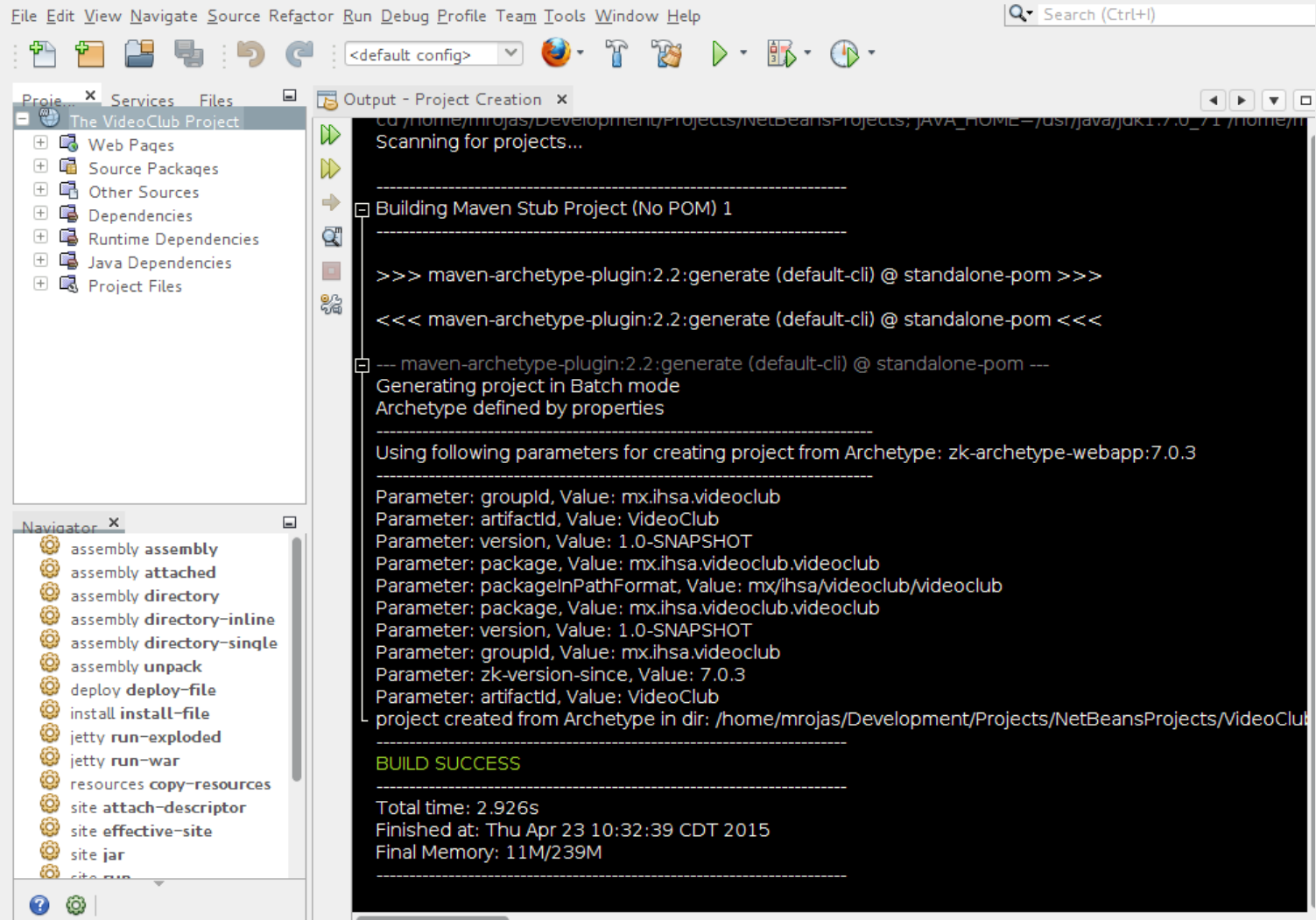
Package: mx.ihsa.videoclub.videoclub (Optional)

Additional Creation Properties:

Key	Value
zk-version-since	7.0.3

< Back Next > Finish Cancel Help

Se crea la estructura y se descargan las bibliotecas de ZK



Dependencias adicionales

- jOOQ
 - jOOQ generates Java code from your database and lets you build type safe SQL queries through its fluent API
- C3P0
 - c3p0 hopes to provide DataSource implementations more than suitable for use by high-volume "J2EE enterprise applications"
- Controlador JDBC de PostgreSQL

Abrimos el archivo pom.xml para agregar dependencias

The screenshot shows an IDE interface with the following components:

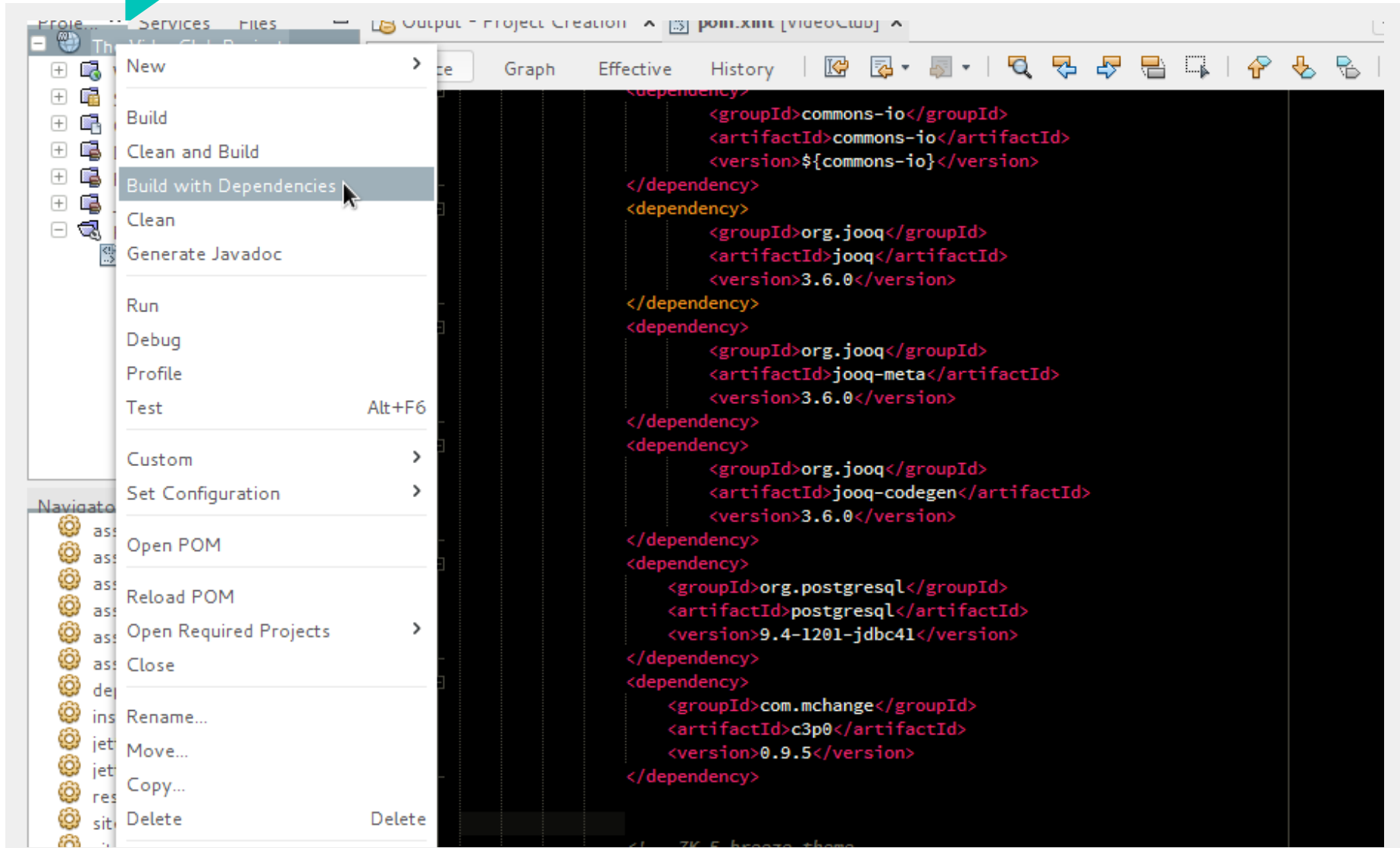
- Project Explorer:** Displays the project structure for 'The VideoClub Project'. The 'pom.xml' file is selected under 'Project Files'.
- Navigator:** Shows the 'POM model' with various fields like 'Model Version: 4.0.0', 'GroupId: mx.ihsa.videoclub', 'ArtifactId: VideoClub', 'Packaging: war', 'Name: The VideoClub Project', 'Version: 1.0-SNAPSHOT', 'Description: The VideoClub Project', 'Licenses', 'Build', 'Repositories', 'Plugin Repositories', and 'Dependencies'.
- Source Editor:** Displays the content of the 'pom.xml' file. The code is as follows:

```
40         </pluginRepository>
41     </pluginRepositories>
42     <dependencies>
43         <dependency>
44             <groupId>org.zkoss.zk</groupId>
45             <artifactId>zkbind</artifactId>
46             <version>${zk.version}</version>
47         </dependency>
48         <dependency>
49             <groupId>org.zkoss.zk</groupId>
50             <artifactId>zul</artifactId>
51             <version>${zk.version}</version>
52         </dependency>
53         <dependency>
54             <groupId>org.zkoss.zk</groupId>
55             <artifactId>zkplus</artifactId>
56             <version>${zk.version}</version>
57         </dependency>
58         <dependency>
59             <groupId>org.zkoss.zk</groupId>
60             <artifactId>zhtml</artifactId>
61             <version>${zk.version}</version>
62         </dependency>
63
64         <dependency>
65             <groupId>commons-io</groupId>
66             <artifactId>commons-io</artifactId>
67             <version>${commons-io}</version>
68         </dependency>
69
70         <!-- ZK 5 breeze theme
71         <dependency>
72             <groupId>org.zkoss.theme</groupId>
73             <artifactId>breeze</artifactId>
74             <version>${zk.version}</version>
75             <optional>true</optional>
```

Agregamos las dependencias

```
<dependency>
  <groupId>org.jooq</groupId>
  <artifactId>jooq</artifactId>
  <version>3.6.0</version>
</dependency>
<dependency>
  <groupId>org.jooq</groupId>
  <artifactId>jooq-meta</artifactId>
  <version>3.6.0</version>
</dependency>
<dependency>
  <groupId>org.jooq</groupId>
  <artifactId>jooq-codegen</artifactId>
  <version>3.6.0</version>
</dependency>
<dependency>
  <groupId>org.postgresql</groupId>
  <artifactId>postgresql</artifactId>
  <version>9.4-1201-jdbc41</version>
</dependency>
<dependency>
  <groupId>com.mchange</groupId>
  <artifactId>c3p0</artifactId>
  <version>0.9.5</version>
</dependency>
```

Descargar las dependencias (Build with Dependencies)



Cómo construir una aplicación programáticamente

- Primero definimos el punto de entrada
 - `index.zul`
- Luego creamos la implementación de nuestra aplicación dentro de un objeto de clase `org.zkoss.zul.Window`
- Agregamos los elementos necesarios de interfaz de usuario y escribimos la lógica de manejo de eventos

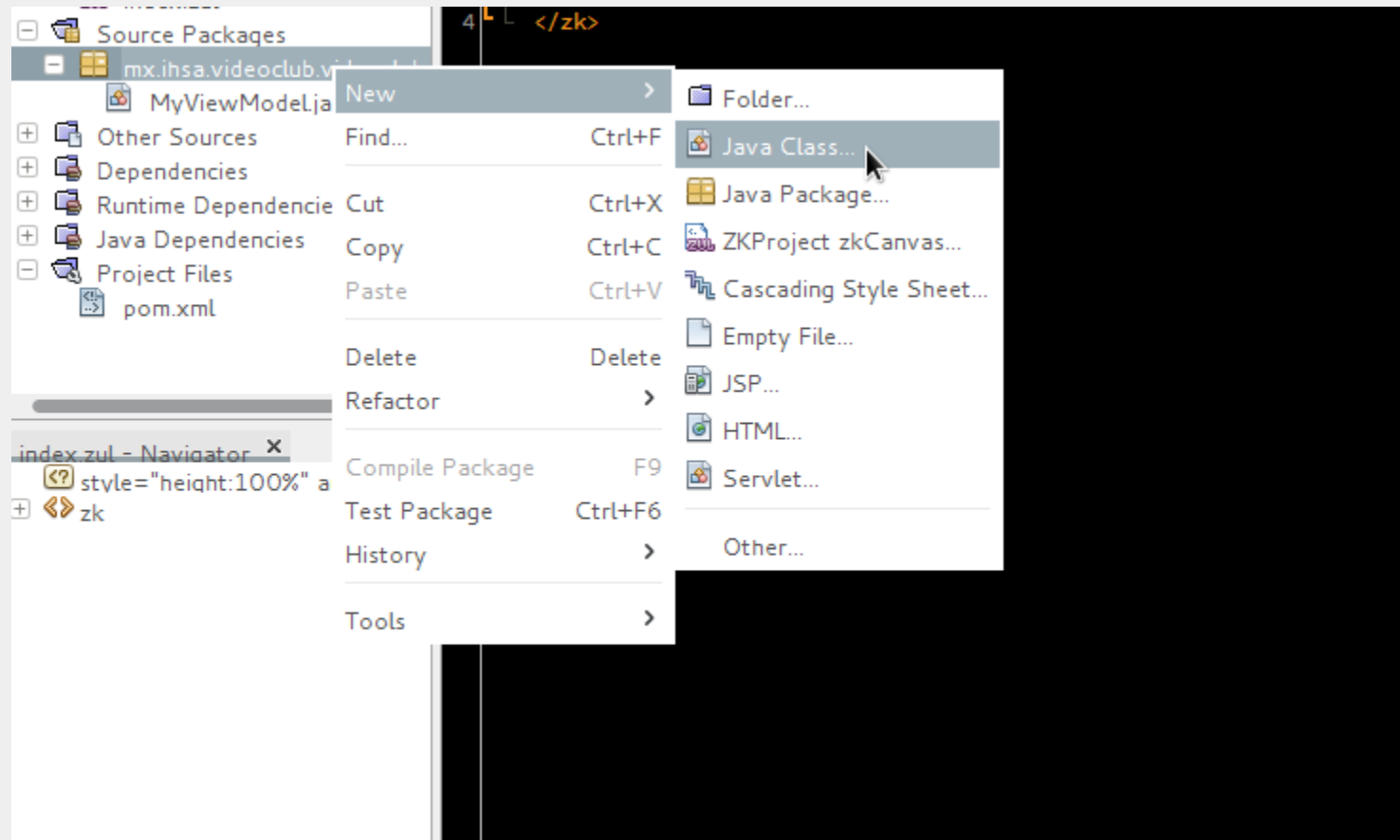
Punto de entrada y ventana inicial

- **index.zul**
 - Simplemente es el punto de entrada
 - Invoca a nuestra clase en Java
 - Quitamos lo que generó maven al crear la aplicación a partir del arquetipo
- **VentanaFilm.java**
 - Ventana de la aplicación
 - Contiene todos los elementos de la interfaz de usuario

Borrar el contenido actual y agregar lo siguiente

```
<?page style="height:100%"
      automaticTimeout="true" id="ihsa"?>
<zk>
    <window use="mx.ihsa.videoclub.VentanaFilm"/>
</zk>
```

Crear la ventana de la aplicación



VentanaFilm

New Java Class

Steps

1. Choose File Type
2. **Name and Location**

Name and Location

Class Name:

Project:

Location: ▼

Package: ▼

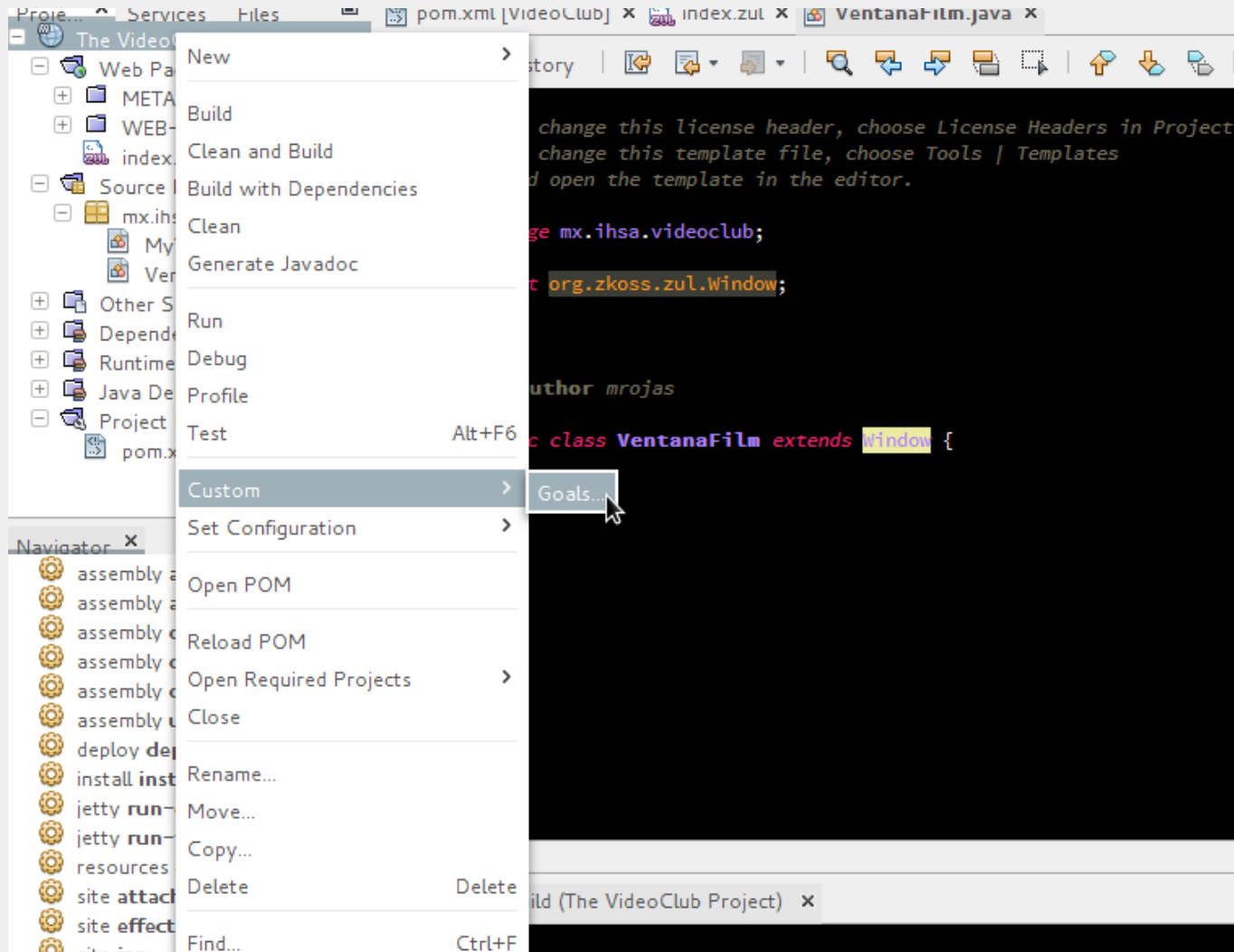
Created File:

< Back Next > Finish Cancel Help

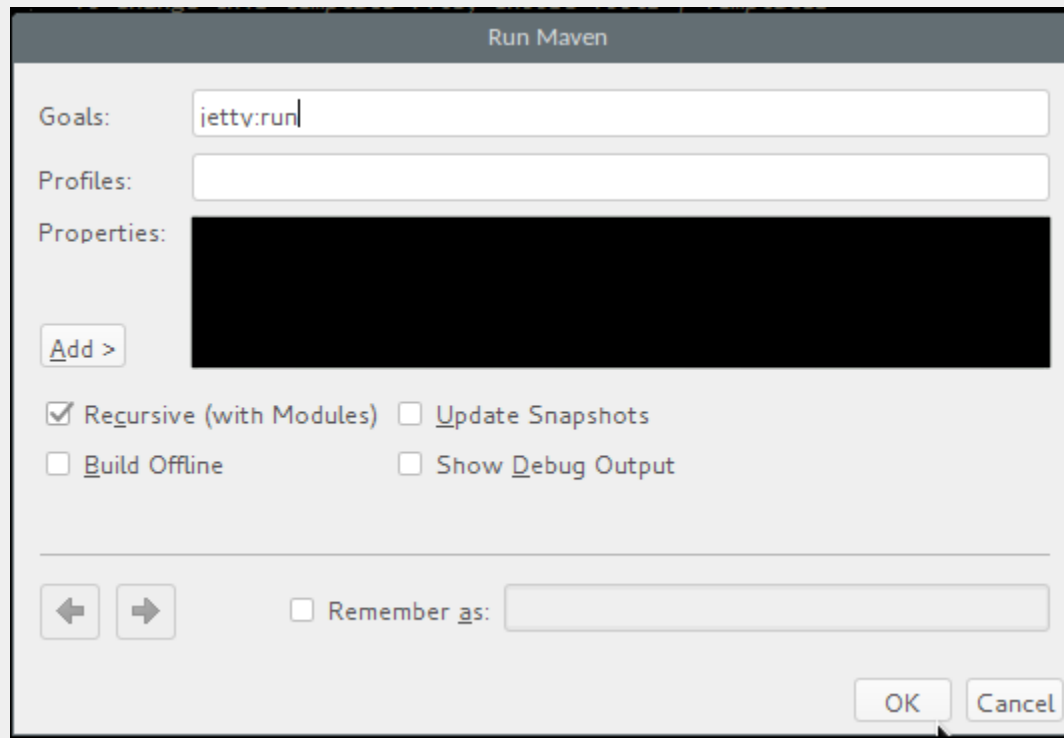
Debe extender de org.zkoss.zul.Window

```
1  /*
2   * To change this license header, choose License Headers in Project Properties.
3   * To change this template file, choose Tools | Templates
4   * and open the template in the editor.
5   */
6  package mx.ihsa.videoclub;
7
8  import org.zkoss.zul.Window;
9
10 /**
11  *
12  * @author mrojas
13  */
14 public class VentanaFilm extends Window {
15
16 }
17
```

Vamos a ejecutar la aplicación para validar



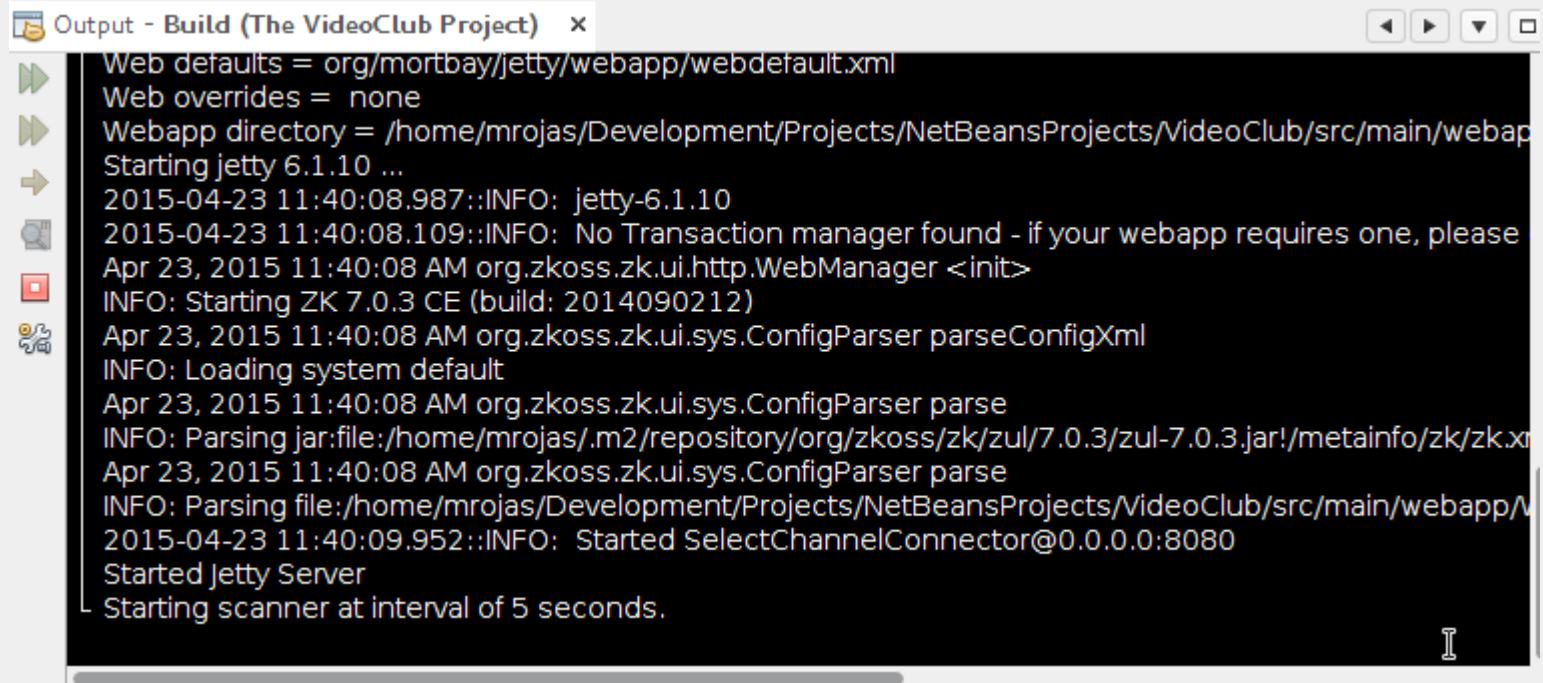
Teclear jetty:run



The image shows a 'Run Maven' dialog box with the following fields and options:

- Goals:** A text field containing 'jetty:run'.
- Profiles:** An empty text field.
- Properties:** A large black rectangular area.
- Add >** A button located below the Properties field.
- Options:**
 - ☒ Recursive (with Modules)
 - ☐ Update Snapshots
 - ☐ Build Offline
 - ☐ Show Debug Output
- Navigation:** Two arrow buttons (left and right) at the bottom left.
- Remember as:** A checkbox and a text field at the bottom.
- Buttons:** 'OK' and 'Cancel' buttons at the bottom right.

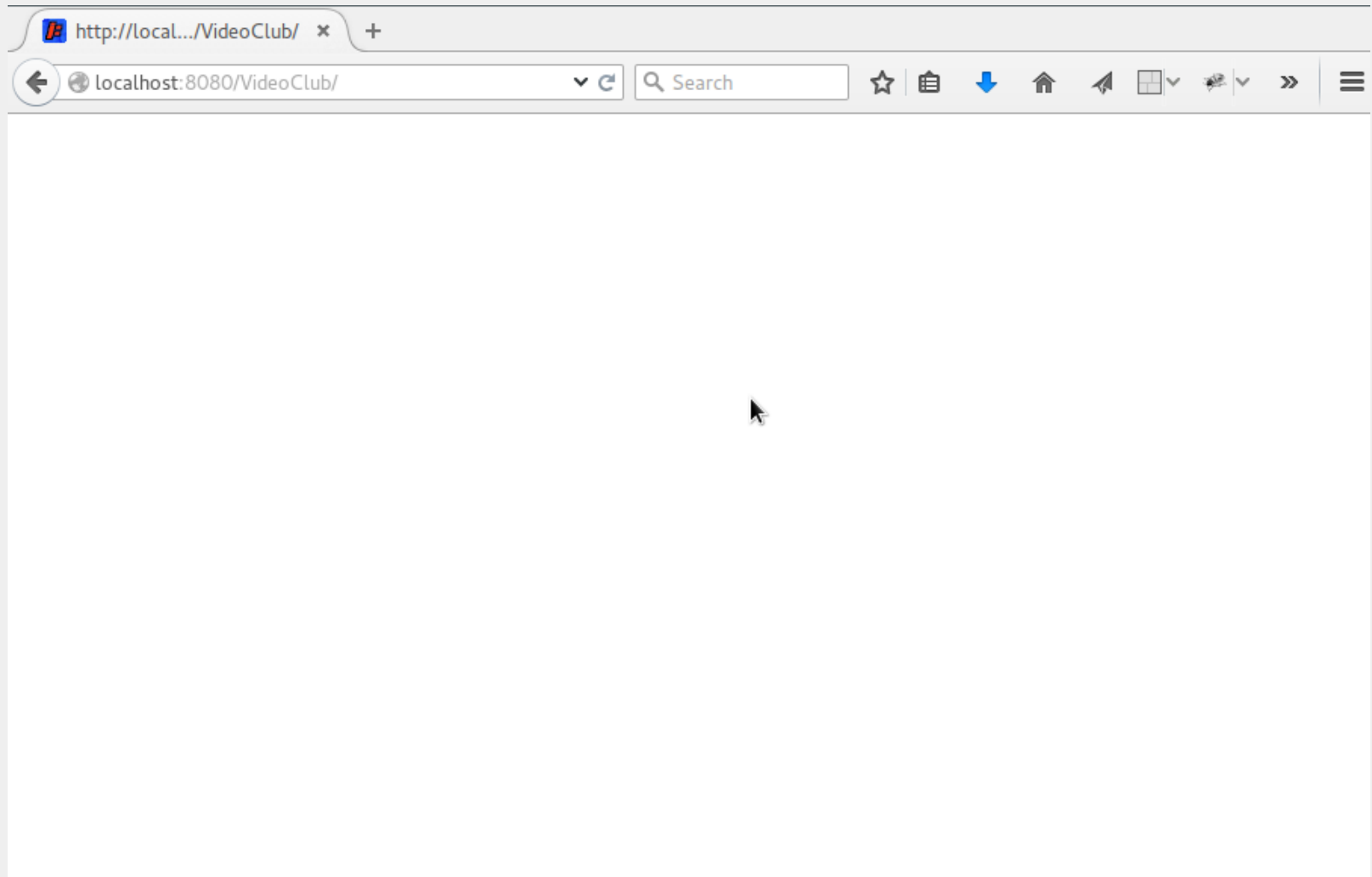
Al aparecer este mensaje ya podemos acceder a la aplicación



The screenshot shows the 'Output - Build (The VideoClub Project)' window in NetBeans. The window has a title bar with a close button and navigation icons. On the left side of the window, there is a vertical toolbar with icons for running, debugging, and other actions. The main area of the window displays the following text:

```
Web defaults = org/mortbay/jetty/webapp/webdefault.xml
Web overrides = none
Webapp directory = /home/mrojas/Development/Projects/NetBeansProjects/VideoClub/src/main/webapp
Starting jetty 6.1.10 ...
2015-04-23 11:40:08.987::INFO: jetty-6.1.10
2015-04-23 11:40:08.109::INFO: No Transaction manager found - if your webapp requires one, please
Apr 23, 2015 11:40:08 AM org.zkoss.zk.ui.http.WebManager <init>
INFO: Starting ZK 7.0.3 CE (build: 2014090212)
Apr 23, 2015 11:40:08 AM org.zkoss.zk.ui.sys.ConfigParser parseConfigXml
INFO: Loading system default
Apr 23, 2015 11:40:08 AM org.zkoss.zk.ui.sys.ConfigParser parse
INFO: Parsing jar:file:/home/mrojas/.m2/repository/org/zkoss/zk/zul/7.0.3/zul-7.0.3.jar!/META-INF/zk/zk.xml
Apr 23, 2015 11:40:08 AM org.zkoss.zk.ui.sys.ConfigParser parse
INFO: Parsing file:/home/mrojas/Development/Projects/NetBeansProjects/VideoClub/src/main/webapp/WEB-INF/zk.xml
2015-04-23 11:40:09.952::INFO: Started SelectChannelConnector@0.0.0.0:8080
Started Jetty Server
Starting scanner at interval of 5 seconds.
```

Accedemos a la aplicación y nos mostrará



No hay contenido aun ...

Antes de agregar contenido y funcionalidad veamos algunos de los elementos que utilizaremos:

- BorderLayout
 - North
 - Center
 - South
 - East
 - West

Consultar <http://www.zkoss.org/zkdemo>

Borderlayout

- Lo utilizaremos para distribuir el contenido en la ventana
- Puede dividirse en 5 zonas
- Cada zona a su vez puede contener elementos básicos u otros contenedores
- Cada zona puede redimensionarse de forma dinámica

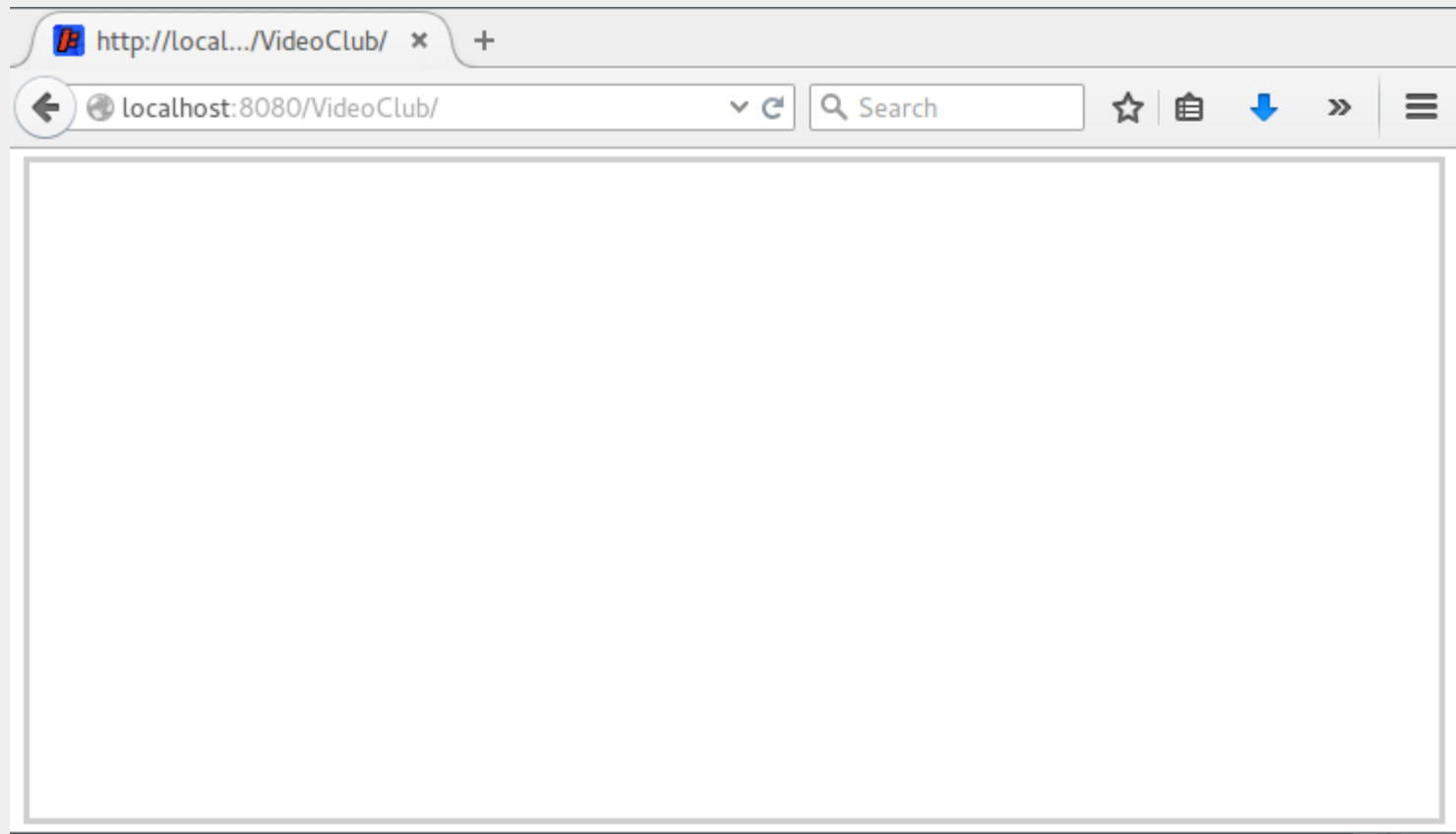
Ejemplo básico

Agregamos los elementos

```
20 public class VentanaFilm extends Window {  
21  
22     private BorderLayout mainLayout;  
23     private North northPnl;  
24     private Center centerPnl;  
25     private South southPnl;  
26     private East eastPnl;  
27     private West westPnl;  
28  
29  
30     public VentanaFilm() {  
31         init();  
32     }  
33  
34     private void init() {  
35         setWidth("100%");  
36         setHeight("100%");  
37  
38         mainLayout = new BorderLayout();  
39         northPnl = new North();  
40         southPnl = new South();  
41         centerPnl = new Center();  
42         eastPnl = new East();  
43         westPnl = new West();  
44  
45         mainLayout.appendChild(northPnl);
```

Y ahora vemos “algo” ...

Aparecen unas líneas de los bordes de los paneles

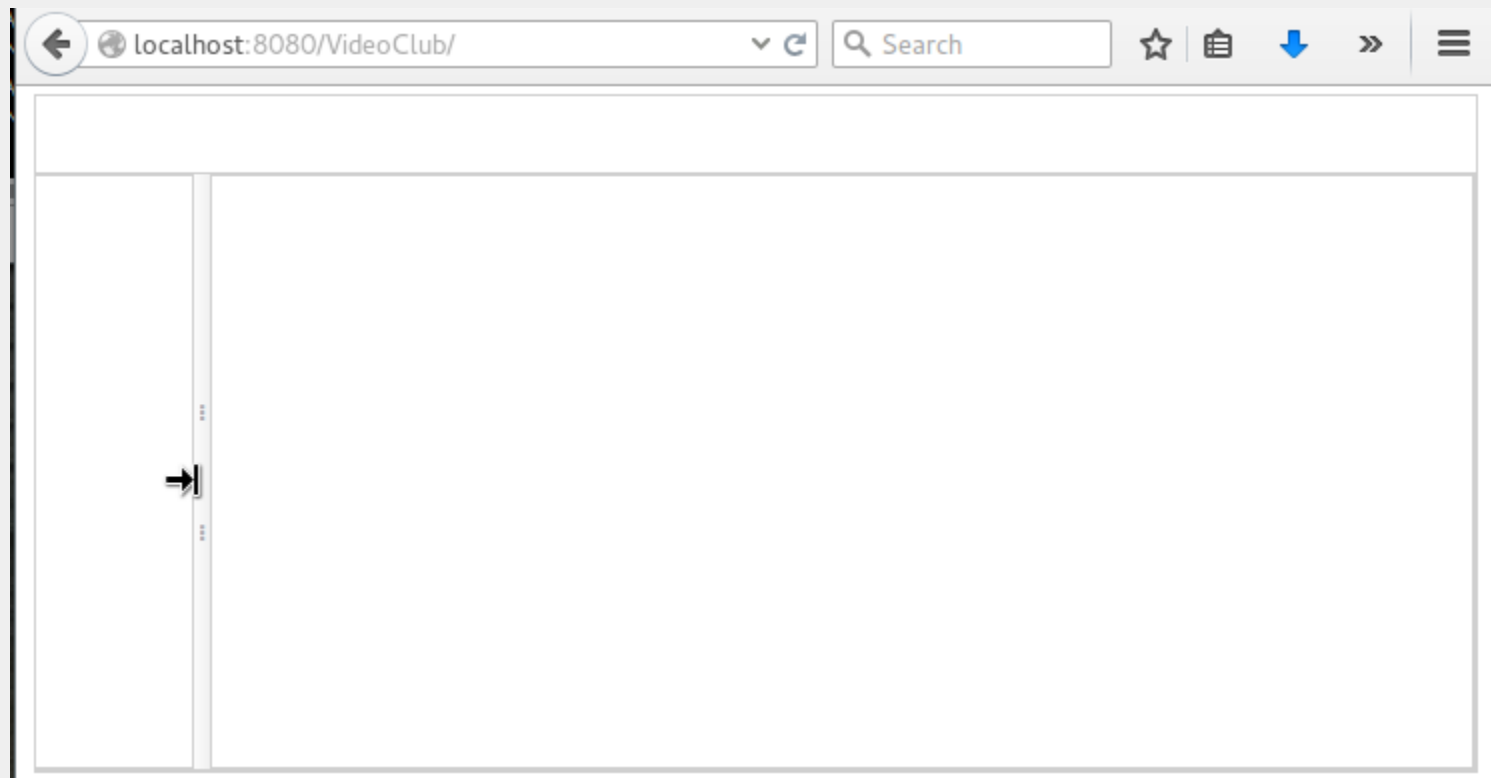


Vamos a hacerlos un poco más interesantes

- Aumentamos la altitud del panel norte
- Aumentamos el ancho del panel oeste y lo hacemos redimensionable

```
38     mainLayout = new BorderLayout();
39
40     northPnl = new North();
41     northPnl.setHeight("40px");
42
43     southPnl = new South();
44     centerPnl = new Center();
45     eastPnl = new East();
46
47     westPnl = new West();
48     westPnl.setSplittable(true);
49     westPnl.setWidth("80px");
50
51     mainLayout.appendChild(northPnl);
52     mainLayout.appendChild(southPnl);
53     mainLayout.appendChild(centerPnl);
54     mainLayout.appendChild(eastPnl);
55     mainLayout.appendChild(westPnl);
56
57     appendChild(mainLayout);
```

Cambiaron los paneles norte y oeste



Vamos a hacerlo más interesante



Agreguemos contenido

http://local.../VideoClub/ x +

localhost:8080/VideoClub/

Search

Video Club IHSA

Categorias

Películas



Ahora es necesario agregar algo de acción ;)

- Los botones no disparan ninguna acción
- No hay acceso a datos
- Ahora agregaremos código previamente generado para reducir el tiempo
 - Clases de modelo de la base de datos del videoclub
 - Clases de utilería

Vamos a hacer que los botones respondan al usuario

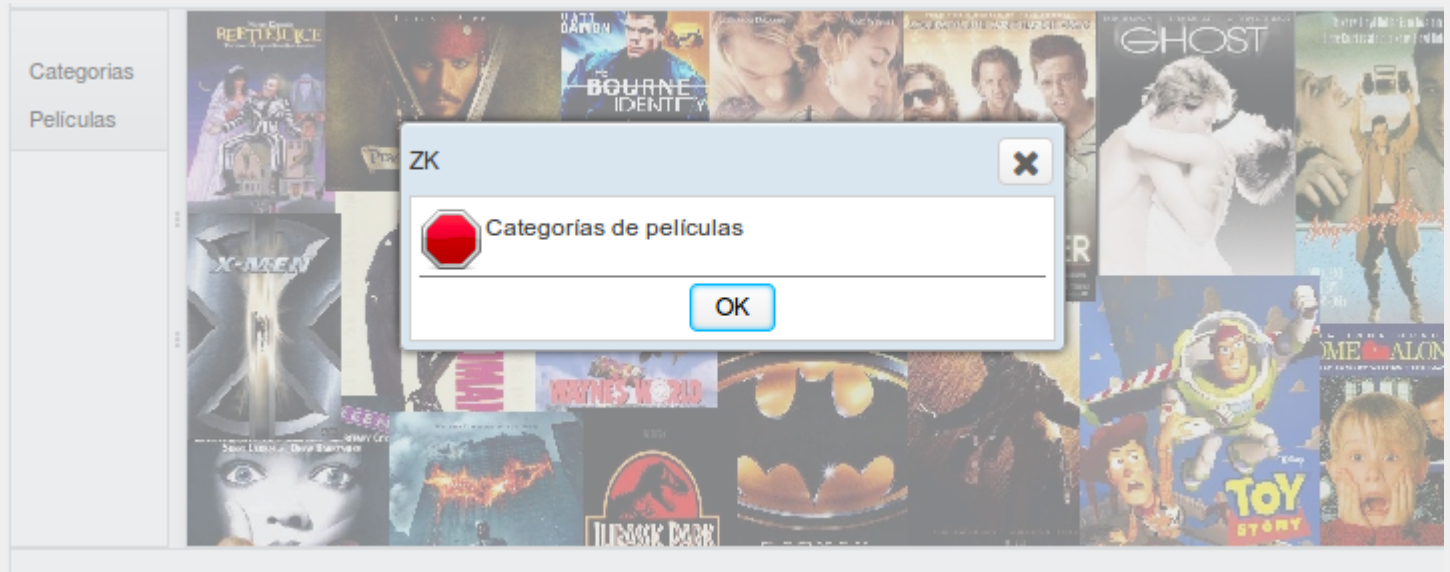
- Los botones disparan eventos
- Es necesario escribir código que atrape y reaccione a esos eventos
 - Un `EventListener`
- Piensa en una aplicación de escritorio, no en una aplicación web

```
Toolbar toolBar = new Toolbar("vertical");
Toolbarbutton button = new Toolbarbutton("Categorias");
toolBar.appendChild(button);
button.addEventListener(Events.ON_CLICK, new EventListener<Event>() {
    @Override
    public void onEvent(Event event) throws Exception {
        // ...
    }
});
```


Vamos a agregarle funcionalidad al EventListener

- Mostraremos un simple diálogo de notificación

```
button.addEventListener(Events.ON_CLICK, new EventListener<Event>()  
@Override  
public void onEvent(Event event) throws Exception {  
    Clients.alert("Categorías de películas");  
}  
});
```

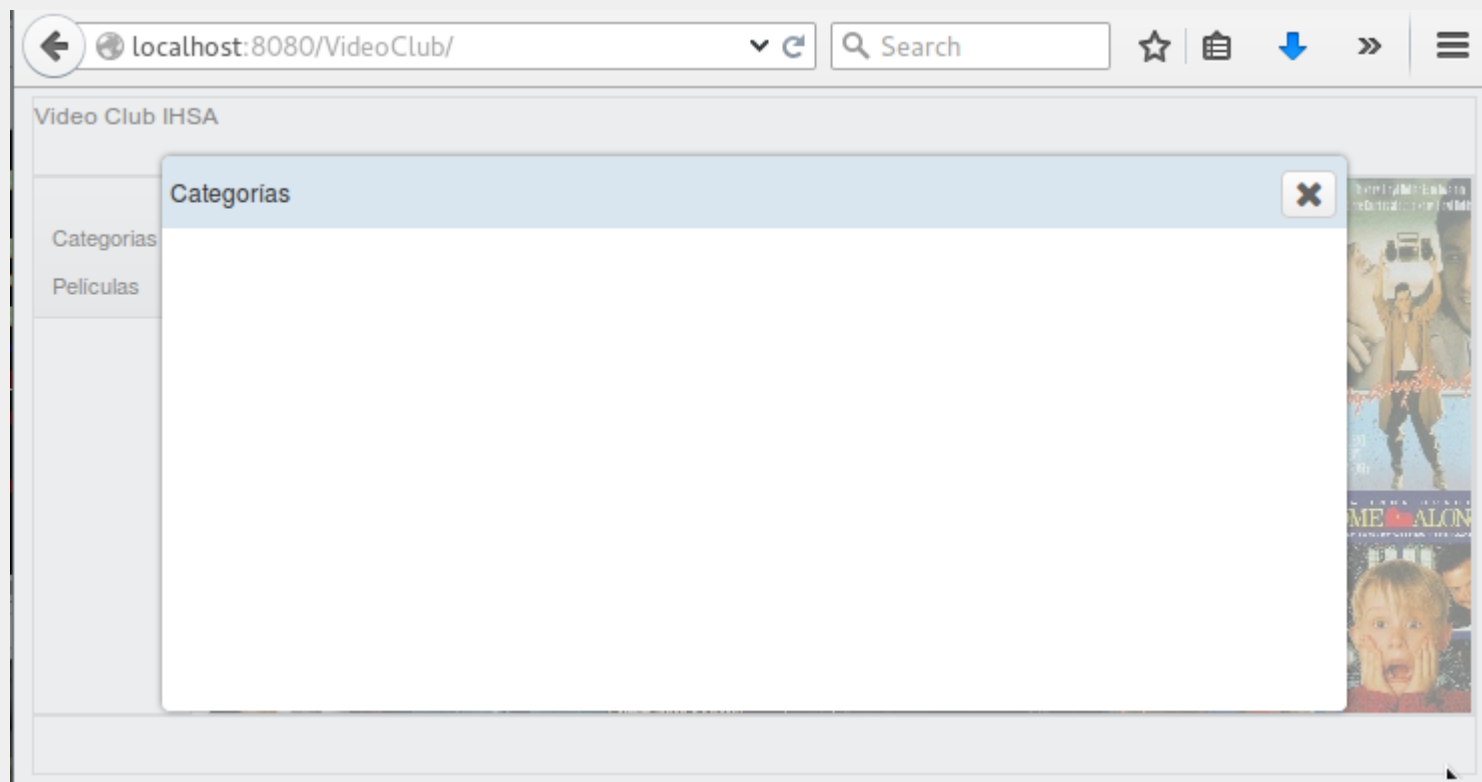


Ahora algo más interesante

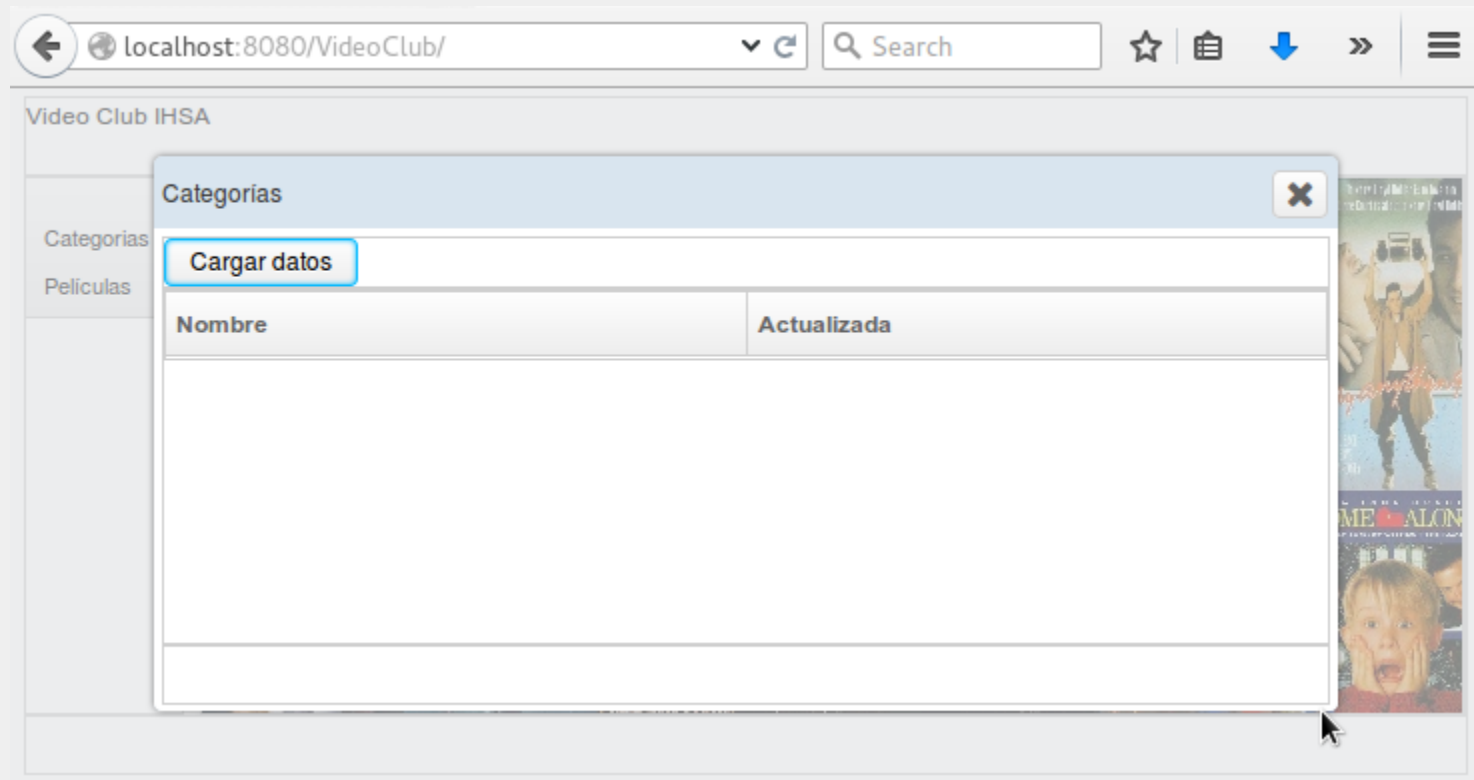
Mostrar un listado de categorías de películas

```
84     Toolbar toolBar = new Toolbar("vertical");
85     Toolbarbutton button = new Toolbarbutton("Categorias");
86     toolBar.appendChild(button);
87     button.addEventListeners(Events.ON_CLICK, new EventListener<Event>() {
88         @Override
89         public void onEvent(Event event) throws Exception {
90             openWindow();
91         }
92     });
93
94     button = new Toolbarbutton("Películas");
95     toolBar.appendChild(button);
96
97     westPnl.appendChild(toolBar);
98 }
99
01 private void openWindow() {
02     VentanaCategory vntCategory = new VentanaCategory();
03     vntCategory.setPage(this.getPage());
04     vntCategory.setPosition("center");
05     vntCategory.doModal();
06 }
```

Ventana creada



Mostremos ahora contenido de la base de datos



¿Y cómo se le hace para que pinte los datos?

Con un ListitemRenderer:

```
categories.setItemRenderer(new ListitemRenderer<Category>() {  
    @Override  
    public void render(Listitem item, Category category, int index) throws Exception {  
        Listcell cell = new Listcell();  
        cell.appendChild(new Label(category.getName()));  
        item.appendChild(cell);  
  
        cell = new Listcell();  
        Datebox lastUpdated = new Datebox(category.getLastUpdate());  
        lastUpdated.setReadOnly(true);  
        lastUpdated.setDisabled(true);  
        cell.appendChild(lastUpdated);  
        item.appendChild(cell);  
    }  
});
```

Mostremos ahora contenido de la base de datos

localhost:8080/VideoClub/

Search

Video Club IHSA

Categorías

Películas

Cargar datos

Nombre	Actualizada
Action	Feb 15, 2006
Animation	Feb 15, 2006
Children	Feb 15, 2006
Classics	Feb 15, 2006




Pero no muestra todos, necesitamos agregar paginación

```
centerPnl = new Center();
centerPnl.setStyle("overflow : auto");
Listbox categories = new Listbox();
categories.setModel(categoryModel);
categories.setMold("paging");
categories.setPageSize(3);

Listhead header = new Listhead();
header.appendChild(new Listheader("Nombre"));
header.appendChild(new Listheader("Actualizada"));
```

Categorias

Cargar datos

Nombre	Actualizada
Classics	Feb 15, 2006 
Comedy	Feb 15, 2006 
Documentary	Feb 15, 2006 

« < 2 / 6 > » [4 - 6 / 16]

¿Y como evito que el usuario le pique a la pantalla mientras se cargan los datos?

- Vamos a tener que simularlo en este caso
- Para bloquear la pantalla se utilizan los eventos en eco
- Para ello modificamos la forma de disparar el evento y de atraparlo

Cambiamos el manejo del evento

- Ahora la clase será la que atrape y maneje el evento

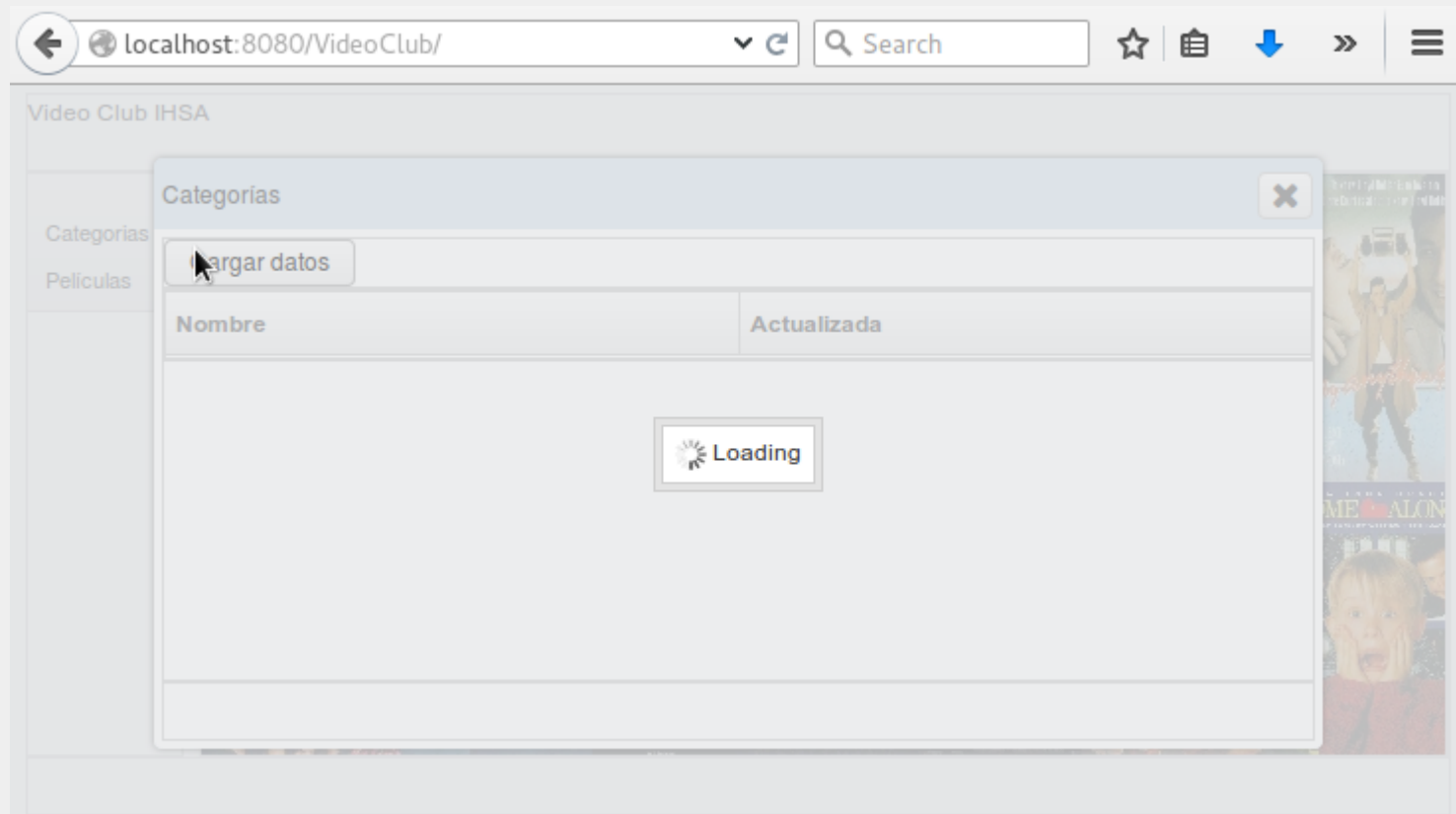
```
public class VentanaCategory extends Window implements EventListener<Event> {  
    private BorderLayout mainLayout;  
    private North northPnl;  
    private South southPnl;  
    private Center centerPnl;
```

```
    Button btnCargar = new Button();  
    btnCargar.setLabel("Cargar datos");  
    btnCargar.addActionListener(Events.ON_CLICK, this);
```

E invocamos el método en eco

```
public void loadData() {  
    //solo para hacer tiempo  
    for(long i = 0; i < 100000000000L; i++) {  
          
    }  
  
    if(!categoryModel.isEmpty()) {  
        categoryModel.clear();  
    }  
  
    categoryModel.addAll(CategoryDao.getAll());  
  
    Clients.clearBusy();  
}  
  
@Override  
public void onEvent(Event event) throws Exception {  
    Clients.showBusy("Loading");  
    Clients.response(new AuEcho(this, "loadData", null));  
}
```

Ahora se bloquea la pantalla



Ahora un ejemplo más completo



Filtrar películas por
categoría

Ahora pondremos en práctica varios conceptos

- Primero renombraremos nuestra primer ventana para que sea el menú principal
- Luego crearemos otra ventana que muestre las películas filtradas por categoría

←

localhost:8080/VideoClub/

▼ ↺

🔍 Search

☆

📁

⬇

»

☰

Video Club IHSA

Categorías

Películas

Películas

Categorías

Action

▼

Titulo	Resumen	Año	Duración
AMADEUS HOLY	A Emotional Display of a Pioneer And a Technici	2006	113
AMERICAN CIRCUS	A Insightful Drama of a Girl And a Astronaut who	2006	129
ANTITRUST TOMATOES	A Fateful Yarn of a Womanizer And a Feminist w	2006	168

⏪

⏩

1

/ 22

⏪

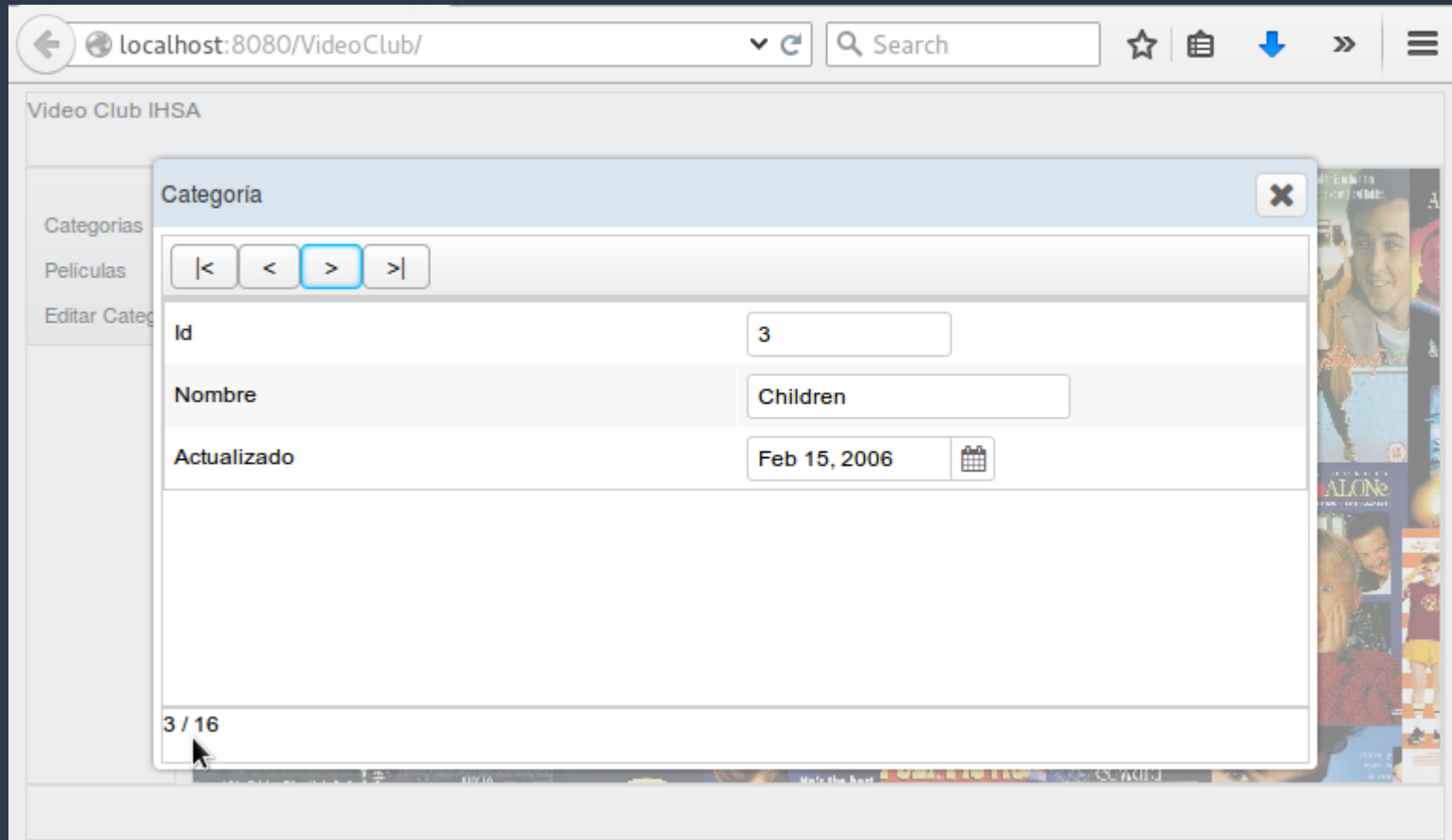
⏩

[1 - 3 / 64]

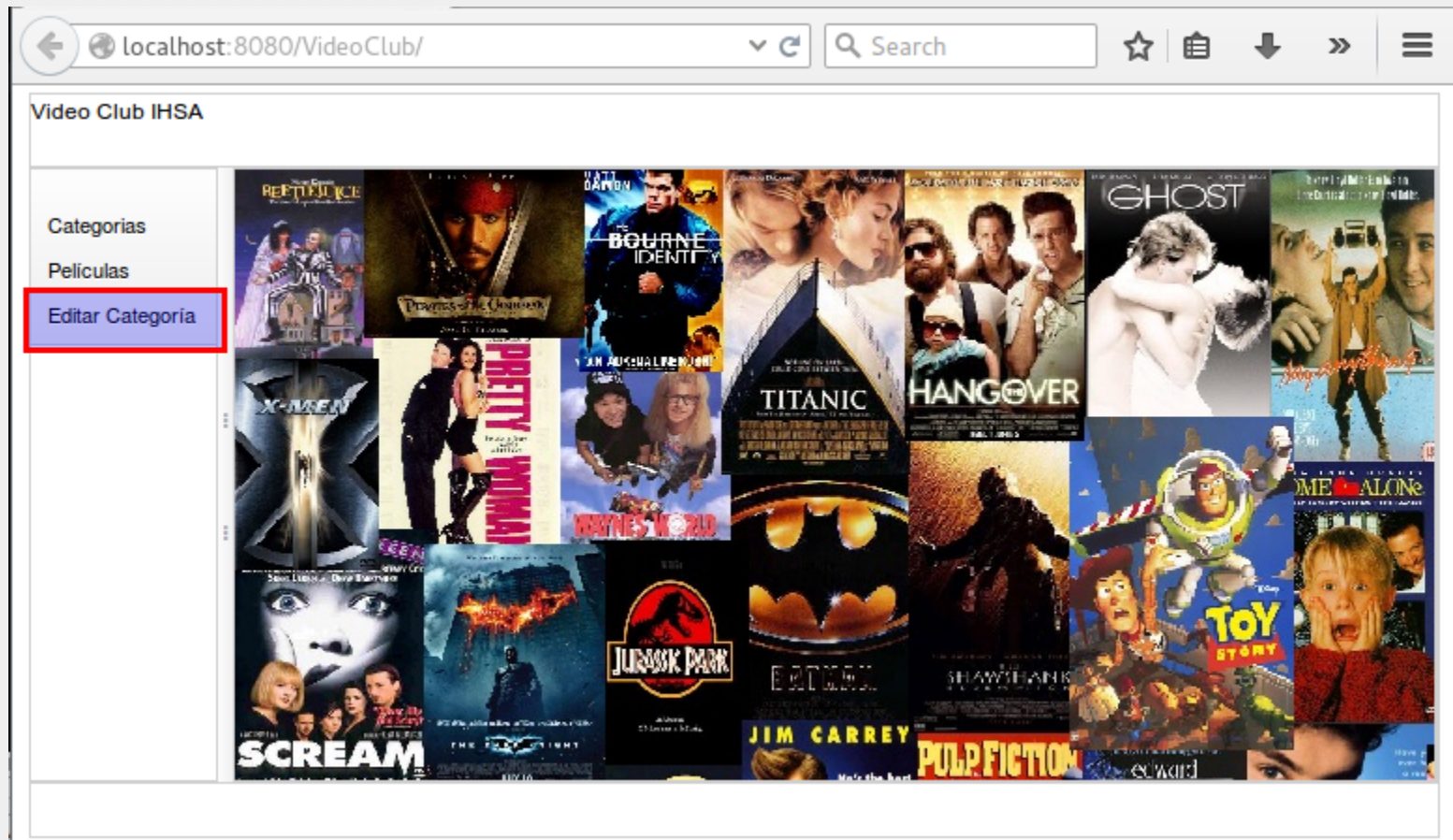
javascript;

Vamos a agregar una ventana para mostrar un solo registro

- Ya tenemos una vista de lista para las categorías
- Ahora mostremos todos los campos de dicha tabla en una ventana a la medida
- Tendremos una barra de navegación, un panel central donde se mostrarán los datos y una barra de notificaciones



Primero tenemos que agregar la opción a nuestra pantalla principal



Luego agregamos los elementos visuales

- BorderLayout, North, Center, South
- Creamos una barra de navegación
 - Toolbar, Button
- Necesitamos atrapar los eventos de los botones para navegar
- Un Grid contendrá los campos de texto a mostrar

Un ejemplo de extensión de funcionalidad

- Extenderemos la clase Comboitem de ZK para que tenga un constructor que acepte el valor a desplegar y el valor interno a

```
14 public class ComboItem extends Comboitem {  
15  
16     public ComboItem(String label, Object value) {  
17         super(label);  
18         setValue(value);  
19     }  
20 }
```

```
private void initData() {  
    Clients.showBusy("Loading ...");  
    List<Category> lstCatego = CategoryDao.getAll();  
  
    for (Category category : lstCatego) {  
        ComboItem ch =  
            new ComboItem(category.getName(), category.getCategoryId());  
        cbCategories.appendChild(ch);  
    }  
  
    cbCategories.setSelectedIndex(0);  
  
    updateFilms();  
}
```

Resumen

- Vimos como crear una aplicación sencilla
- Con manejo de eventos
- Con acceso a base de datos
- Con personalización de “pintado” de datos
- Consejo : consulten la página de demo de ZK, así como se llaman las etiquetas ZUL de los componentes, así se llaman las clases en Java

Pendientes ...

- Reutilización de código
- Manejo de un modelo de datos
- Creación de componentes propios