This video will give you an overview of how IBM ILOG JViews use the Java Server Faces technology to build web applications.

To make the best use of the following, it is advised to have some basic knowledge about JSF.

And if you want to start a web project as shown in the second part of video, you will need to have JViews installed and a Java IDE.

But first, let's have a look at the JViews Faces library.

This charts shows too parts: the browser side and the server side. The communication between these two entities is done through picture and JS.

Each time the client sends a request, he receives a new bitmap of the JViews Application. This image is not totally. Thanks to the server side JViews component the user can select, navigate, zoom, edit.

The bitmap is rendered by the server, and to be more specific, by the JViews Rendering engine that is hosted by the web server. The engine is itself connected to a datasource to populate/update the application model.

Now let's focus on the communication part. With the traditional HTTP protocol, the application used to remain inactive from the request to the server response. (I am sure that all of you once got mad at the hourglass pointer that sometimes would never go away).

Now with the Ajax approach (based on asynchronous requests), the user keeps the control of the application even while the server is processing the requests. This results in smoother and more responsive applications.

Now that we know the big picture, let's dive in the code with this chart showing the keys elements of a JSF application based on JViews.

In the jsp page, you need to "import" the tag libraries containing the JViews components definition. Then it's possible to create JViews components and fill their parameters.

For instance, with that button, we define the behavior of the click event with some JS code.

For the map view, we define the underlying data thanks to a backing bean called diagramBean. This backing bean is declared in the jsf configuration file and defined in a java class hosted by the server.

It terms of technical requirements, the JViews Faces library works with the standards/ and most popular versions of third party software.

Java 5 and 6, JSF 1.1 and 1.2.

The product has been tested with portals such as Pluto 1.16 among others.

We support all the mainstream application servers, in particular IBM Websphere Application Server and Tomcat.

Two latest versions of IE and Firefox are OK.

The last release of JViews have had a particular focus on compatibility with most popular Ajax libraries such as IceFaces, Trinidad, Rich Faces, MyFaces or Dojo.

As said before, portals are also welcome.

To develop your JViews Faces application you can use any type of java environment. IDE's such as Eclipse and its extension are of course supported.

The second part of this video is more practical. We are going to create a simple JViews web application. For that we will use IBM Rational application developer.

Let's go through the integration steps.

1) Our first job is to pick and declare the runtime environment that is the web server that will host the application.
2) Then, we can create a new web project with the right settings
3) Last part is coding : JSP, java and JavaScript. In that simple example we will only do jsp coding.

As application server, we will use the Tomcat distribution that is bundled with JViews. It's located in the JViews Framework folder.

Once this is done, we can create our new Web Dynamic project.

The runtime is Tomcat, and we do not need any EAR.

We need to check the needed libraries for JSF within the Faces Project panel.

Then, from the JSF Library Panel we've got to add the JSF implementation libs. In our case, we just pick the ones that are shipped with JViews.

Till now, it's been like any JSF application. Now we're going to focus on the JViews specific part, where the configuration files hook the JViews classes. The mandatory bindings can be found in the documentation: it's the JViews faces controller.

No need to write it manually for we can find it in one of the numerous samples given with JViews.

A few edits are necessary like the application name and the welcome page.

The faces config file remains unchanged because we have no backing bean or navigation rule in that example.

Before getting any further, something we need is to add the JViews libs to the classpath project. When doing so, we indicate that the JViews libs are part of the dependencies of the web project by clicking those check boxes.

Once that step is cleared, we can take care of the data that will feed our little example.

For that we will just pick the data set of one of the numerous code samples of the product.

Now we can dive into the real code and create a new JSP page.

Now what we want to do is to insert the JViews faces components into that page.

To do so, we first include the taglib direction at the header of our page.

We will just use two tags : diagrammerView and selectInteractor. The components are empty. So we have to fill the parameters.

For instance, I am now filling the data field, that enables us to retrieve the datasource to populate the diagram.

The come the size, the id, the data.

This is can be done pretty fast from the properties panel.

Last step is now cleared and we should be able to test our JViews faces example by running the main jsp page within the tomcat server.

The result first shows up within Eclipse. But if we copy paste de URL, we can also run through external browsers.

Internet Explorer and Firefox.

Here you go !

All’s left is to thank you for your attention and to wish you all good luck with your JViews Faces project.

Bye bye !