INTRO

Hello and welcome to this fifteen minutes presentation showing how to build a JViews Map Display for a desktop application.
This presentation will show you how to build a mapping application in few lines of codes by taking advantage of JViews tools and API.
Before to watch this presentation, it's preferable to have a look at 2 others presentation.
The first video, named "Map Tool Chain" will give you an introduction to the usage of the Map Builder allowing you to build easily a map background.
The second video "Diagram Display Desktop" is dedicated to the Diagrammer product, a Model View Controller implementation allowing to create rapidely animated symbols linked to a back-end.
In order to have a better code understanding, it's preferable to have some knowledge of Java and Swing development. We will use Eclipse, even if any other IDE can be used with JViews.
So, an Eclipse background can also help concerning the IDE configuration step.

Here are the different steps that will be presented.
The first step is a pure Swing application development. We will see how to build a Swing display using a JFrame.
The second step shows how to add a Diagrammer view to the frame. This view will load a project that has been build using the Diagrammer Designer.
The third step shows how to add interactivity to the view using a tool bar.
Finally, the last step shows how to add a symbol on top of the map using latitude and longitude information.

First Step: How to build a Swing Application using Eclipse

STEP 1

First of all, we create a New Java Project named SimpleMapApplication.
We check "Create project from existing source" even if we will create the source code.
By this way, we decide the location of the workspace directory. We choose the same name than the project.
Now, we create a new java class named SimpleMapApplication. We check the box allowing creating the main method.

The first step is to create the main window. So, we instantiate a JFrame.
By clicking on the left border, Eclipse propose to choose the JFrame import coming from the Swing package.
Now, we call "setDefaultCloseOperation" method with Exit on close parameter in order to be sure that we really quit the application when closing the window.
The next step is to call the pack method in order to adapt automatically the size of the frame to the subcomponents.
Finally, we call setVisible to show the window.
We can quickly test the result by choosing "Java Application" in the Run AS dialog.

The result is an empty window with just a title bar.

STEP 2

First of all, you can see that we added some files in the data directory.
By double-clicking on the world.idpr file, the diagrammer designer open this project.

The "Background Map" option allows selecting the IVL file that will be used as map background.
By selecting the node leaf in the Style Rules tab, you can choose the palette and the symbol to render a node.
In the "Diagrammer Edition" mode, you can access the model, and edit for example node properties like the type.
By changing the type from 2 to 1, you can see that a style rule is applied and the node picture is changing from a plane to a bus.
It's also possible to add new nodes.

To have a more focus presentation of the designer, I invite you to watch the "Map Tool Chain" video.

Back to Eclipse and java development.
The first step is the IlvDiagrammer instantiation. It's a JComponent extension used as a graphic container.
By right clicking on the left border, you can see the JViews package are not accessible.
We will edit the project properties, in order to add the JViews jar files to the project libraries.
The first jar file to add is jviews-map-sall.jar containing all the JViews Maps API.
The next file is jviews-digrammer-all.jar for the JViews Diagrammer API and jviews-palette-shared-symbols.jar containing the material for the node rendering.
Finally, you also have to add jviews-framework-all.jar containing all the core JViews graphic API.
Now, it's possible to import the IlvDiagrammer package.

The next step is to set a size for the diagrammer view. You can access it using getView() and call the setSize method with the good width and height parameters.
Now, we load the diagrammer project by using setProject(). IlvDiagrammerProject need an URL to access the world.idpr file that has been generated using the designer.

We can add the diagrammer component to the center of the frame using the add method, and the BorderLayout.Center parameter.
We finish this step by calling fitToContents() method on diagrammer. Obviously, it allows resizing the view in order to make all the graphic objects visible.

By launching the application, you get an error message concerning the JViews license because the keys.jlm file is not accessible.
To remove this error message, you can edit the project Run Configuration by adding a path to this file in the Classpath tab.

Now, you obtain the Diagrammer view inside the JFrame window. You will be able to interact with this view by following the next step.

STEP 3

To add some interactions, we will use the JViews maps Manager View Control Bar object which is a JComponent bean object.
It's a pre-packaged object containing all the predefined interactors like zooming, panning or object selection.
It also contains a lot of other interactors that we will detail during the test demonstration.
The first step is obviously to instantiate the IlvJMapsManagerViewControlBar.
Then, we attach this control bar to the diagrammer view using controlbar.setview.
And, finally we add the control bar to the top of the frame using BorderLayout.North parameter.
And, it's done.
Now, we can test the application.
First, you can see the ControlBar is a floatable object. You can dynamically move the position and the attachment of the control bar.
By pressing a toolbar button, you activate a specific interactor.
The arrow button is the select interactor. The default behaviour allows selecting and moving a node on top of the map.
The pan interactor allows scrolling both horizontally and vertically using the mouse.
This interactor allows selecting a rectangle area and zooming on it.
It's also possible to directly zoom or un-zoom.
We provide a magnifier allowing zooming locally inside the normal view.
You can also rotate the view.

STEP 4

The aim of this final step is to show how to add a new node on top of the map using geographic coordinate using the diagrammer API.
For example, we will add a plane on the top middle of the US Map.
First, you have to create a node by calling diagrammer.createNode. The parameter provides the tag value. It returns an object.
We set some properties using setObjectProperty. The first property is the object type. If we use 1, the node image is a plane.
We set the node position using latitude and longitude properties.
First the latitude information gives the north south position. With, 48DN, we are near of the top of the United States.
Then, the longitude information provides the east west position. With 105DW, we are at the center of the United States.
Now, we just have to add this node to the model using addObject. As, there is no parent hierarchy, the second parameter is null.

Thank you.

I hope this quick video helped you to understand the JViews Maps API.
I invite you to download the sample source code and realize some modification to start your first JViews Maps project.
If your interest is more on web application, go to the JViews Maps JSF video to see you to develop the same kind of application for the web.