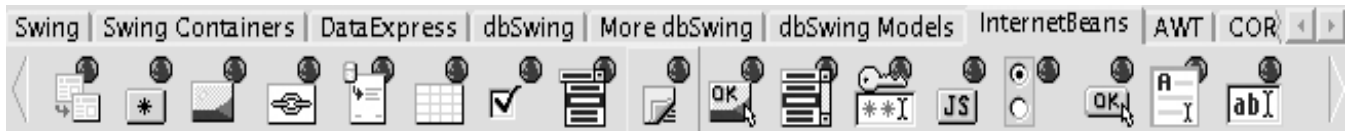


Mr HAKI's JBuilder Jar



InternetBeans Express

Developing a web application in Java isn't too hard. The Java 2 Enterprise Edition contains the Servlet API and Java Server Pages (JSP). The Servlet API contains classes to get data from and to the user's web browser. The Java Server Pages have been introduced to provide a template-like structure for web applications. Before we had JSP we coded the HTML in our Java servlet classes. You can imagine what happens when the web application's look-and-feel needs to be changed. All Java code had to be scanned and modified. With JSP we can leave the look-and-feel in an HTML file, which can be edited by web designers, and code the business logic in the servlet classes.

But JSP 1.0 has a downside: it is too easy for a developer to put all the Java code in the JSP file. And that approach results in a difficult to maintain web application. JSP 1.1 provides developers with a solution for this problem: custom tags. We could define our own tags which would be included in the JSP page. These tags were replaced by Java code written by the developers and the web designers only had to know the syntax of the tag.

Borland introduced components like the PageProducer in Delphi a couple of years ago. The PageProducer was coupled to a template file, thus separating the user interface from the business logic. Now with JBuilder 4 we got the Java equivalent called InternetBeans Express. InternetBeans Express consists of a set of components we can use in the JBuilder Designer to couple HTML templates and their contents with business logic written in Java. We can connect to databases and use live data. And the good part is these components are also implemented as custom tags, so they can be used by JSP 1.1 web applications.

A closer look

The main component of the InternetBeans Express architecture is the IxPageProducer. This component is linked to a static HTML file, which is the template. Then we get a couple of components, like the IxInputField, IxPassword, IxSubmitButton, which need to be coupled with their HTML counterparts in the HTML file. When the application is running these components will contain and serve the data.

The IxTable component is used to display data in a tabular form. For example, the contents of a table from the database. Then we have some useful components to create dynamic links, images and dynamic data not coupled to <FORM> elements.

To use these components we follow these basic steps:

1. First we create the HTML template file. This file contains for example <FORM>, <INPUT>, <TABLE> and tags.

2. If we want to use data from a database, we create a data module.
3. We create a servlet using the Servlet Wizard in JBuilder.
4. We drop the IxPageProducer component in the servlet in the JBuilder Designer. We assign the component to the HTML file.
5. We drop IxControl components in the servlet for every <INPUT>, <TABLE> or element found in the HTML file.
6. We connect each component's dataSet property to a dataset from the data module. We can also assign the columnName property so the correct data from the dataset is shown.
7. Write event handlers for the buttons found on the page, to handle the POSTing or GETting of the page. We also can change the servlet's doGet() and doPost() method to use the IxPageProducer component.
8. In JBuilder 4 we can run the application from within the IDE, by right-clicking on the HTML file and selecting the Web Run option.

Conclusion

The InternetBeans Express architecture provides a good distinction between the user interface and the business logic of an application. Web designers can create the user interface and Java developers can write the business logic. This way everybody is doing what they are good at. Also the fact that the components can be used as custom tags in JSP 1.1 is a big advantage, because it complies with J2EE.

A downside is the documentation (or lack of it) of these components in JBuilder. There is a small tutorial to get us started, but to use all the components to the maximum we have to learn by trial-and-error.



Hubert A. Klein Ikkink is a knowledge engineer for Everest Delphi OplossingsCentrum in The Netherlands. He's a true Java/JBuilder expert, and co-webmaster/contributor for Mr.Haki's JBuilder Machine at <http://www.DrBob42.com/JBuilder>.