XML

UI Design with Java and XML Toolkits
Contributed by Dan Wellman
2006−09−13

XML has revolutionized application UI design in recent years. With a cunning blend of XML and script languages such as JavaScript, rich, aesthetically pleasing applications can be quickly constructed with ease. We've looked at Widgets and XUL as two examples of this in the past and now, I'm going to take a look at some of the innovative Java UI toolkits that implement XML as an integral mechanism for application II design. Please note, this is the first part of a two−part article.

A quick trip to Google shows just how many Java XML UI toolkits there are around now. Many of these toolkits are open source, which is great news for developers, and for people that just want to get involved or to learn the basics of application design. There are also a growing number of full−fledged applications designed to give you an interface of your own in which to design and produce interfaces. The fact that these applications exist at all show that Java and XML GUIs are the answer that many developers have been looking for.

In addition to Java/XML UI toolkits, other specifications are also being developed to make use of XML in GUI development, like XAML (eXtensible Application Markup Language), a language that defines UI objects in XML and renders them using the WPF (Windows Presentation Foundation), which is the new presentation API in the .Net framework 3.0. So, there are quite a few different UI toolkits you can choose from. Whatever associated technologies you have experience with or want to gain experience in, there is bound to be a toolkit or IDE you can use. Make sure you've got the latest Java Development Kit (JDK) rather than just the standard Java Runtime Engine (JRE) before attempting to use any of the UI toolkits.

It would seem to be the next logical step for people that have comfortably used XUL to create add−ins for Mozilla or FireFox, or even mini desktop applications. Whereas XUL must mostly run within either browser, Luxor is an embedded XUL engine that is combined with Java to produce rich desktop applications, including the very popular LimeWire peer−to−peer program. Because specifying the user interface in XUL is so easy, constructing a functional and attractive interface can be done very quickly −− much quicker, and with greatly simplified code in fact than in writing the whole application in Java.

You can also make use of several of the main attractions of XUL; CSS styling which allows a simple mechanism for creating a visual theme for your app, and the skinnability that many users desire, as well as the localizational properties of XUL. JavaScript can be utilized effectively for the simple functions of an interface, but Java gives you more power than Javascript and has been claimed as being easier to learn with a cleaner syntax. JavaScript is also less appropriate outside of a browser environment.

For those of you that aren't interested in learning Java, there is also Luxilla, which is a Luxor development environment you can use without knowing/coding a single line of Java. Luxor is free, open source and released under the GPL. A Luxor web site does exist, at http://luxor−xul.sourceforge.net/index.html, but don't expect anything more than a very basic site. There is little real information for anyone wanting to get started with Luxor, and no introductory tutorials that guide you through the installation process or use.

Luxor comes in the form of an executable .JAR file, much like most of Mozilla/FireFox, which contains the swing class files used to render the UI objects on screen. Luxor enthusiasm does seem to be at a low ebb at present, the latest distribution being around 18 months old. I have to admit, in the time I allocated to playing around with each of the Java

XML toolkits I looked at, I didn't manage to get Luxor working at all.

Because XUL is such an easy to use subset of XML, there have been a lot of projects to incorporate XUL into the Java platform. One of these projects is: JXUL, designed to create a cross−platform execution engine for XUL, so that it can be used without the need for Mozilla to run, or even be present, on a user's system. Nothing seems to have happened with this project for some years now, and it seems to have been superseded by Xulux, although very little information exists for this, so what has happened to it now I couldn't say.

SwiX^{ml} is a small GUI generating engine, released under the Apache Software License that uses plain XML files to specify UI components. It's just 40KB in size, is fast and focuses on just generating GUIs. The XML files are parsed by the generator at runtime and the UI elements described are rendered into javax.swing objects. You still need to have knowledge of swing (which means you have less to learn if you already know it), but you are able to completely separate the GUI construction from the program logic (which as we all know is a good thing for programmers of any discipline to do).

SwiX^{ml} was created by Wolf Paulus in 2003, and I'm guessing that no development has been done for a little while as the copyright statement on the SwiX^{ml} site reads 2003 − 2005. Nevertheless, you can download version 1.5 and install it relatively easily. I found that before it would install correctly, however, I had to add a new environment variable to set the JAVA_HOME variable.

To do this on a Windows XP platform, open the control panel, open the System applet, go to the Advanced tab and open Environment Variables. Click the new button and in the Variable field type JAVA_HOME and in the Value field enter C:Program FilesJavajdk1.5.0_08 provided that is where Java is installed (this is separate from the addition to the existing Path variable that you may also have set at some point prior to using SwiX^{ml}). Once this is done, you can unpack the SwiX^{ml} zip file and double click the build.bat file to install it, which will create the SwiX^{ml} jar file library and you can begin.

JAXX (it doesn't actually stand for anything but you can guess at a glance that Java and XML are involved) is probably the best toolkit in terms of documentation; there is an excellent, well produced, well maintained and most importantly, recently updated web site at http://www.jaxxframework.org/ which gives you everything you need to know to begin creating UIs with JAXX. There is a thorough installation guide and some excellent tutorials to get you started and a live forum for if you get stuck and need some assistance.

One of the things that makes JAXX so useful is its powerful CSS support which enables the skinning of applications via simple style sheets. It is very easy to get started with JAXX; I was able to run through the tutorials and get results on screen very quickly. My experiment GUIs didn't actually do anything as there was no underlying Java to make things work, but it was good just to get something to actually launch. It's easy to look through the example applications provided to see what is going on and how to use the different tags effectively. Out of all of the toolkits I looked at, this was without doubt the easiest to begin using.

Thinlet is another well know Java and XML UI toolkit that differentiates from some of the previous examples by avoiding swing altogether and using the abstract window toolkit (AWT) instead. AWT is an older set of UI designing widgets that swing was developed to replace. One of the key differences between AWT and swing is the nature of how each toolkit renders the various GUI elements, with swing providing more consistency across any platform as opposed to AWT relying more on the underlying OS implementation.

What Thinlet does is provide an extremely small toolkit that excels at creating relatively simple interfaces, when the complexity of some swing components can be avoided. The documentation for Thinlet is quite good, providing a local copy of what can be found on the Thinlet website (www.thinlet.com), including details of the widgets themselves and the API. There are some examples included in the documentation (including an excellent Amazon browser) that you can fire up and look at to get an idea of how things work.

Thinlet is free and open source for anyone from enthusiasts to commercial enterprises and was released under the GNU Lesser Public License. There are some extensive Thinlet tutorials out there, but the documentation included requires a working knowledge of Java and presumes familiarity with class compilation, CLASSPATH variables and more so this is probably not best suited to newcomers to Java application programming. People that already have a working knowledge of Java may be attracted to Thinlet's simplicity.

These are just some of the available resources out there for building XML based GUIs for your Java applications, providing an easily maintainable application with a completely separate presentation layer. You can save valuable development time this way, and there are a variety of toolkits you can use depending on what suits you and your experience level.