# Another Perspective

## GOOGLE'S LOVE AFFAIR WITH IBM'S OFFSPRING

Imagine what would happen if Google and IBM wanted to help you build your web. Now suppose the Googlers and IBMers created free site development tools that were aggressively Open, capable of running on just about any platform and able to build webs that could be used by clients as small as a mobile phone or as large as an engineering workstation. Now imagine that all this has been around a few years and you've pretty much missed it, or at least not appreciated it. Maybe it's time to say "Hello, World" to the Google Web Toolkit and Eclipse.

Before you get too excited, we ought to tell you that none of the webs you are likely to build with the Google Web Toolkit (GWT) and Eclipse (an independent offspring of IBM's Ottawa lab) will be as lean as well-written hand-coded web pages, nor as truly original. On the other hand, the Google and Eclipse development tools can help you build and maintain web sites with catchy features that are widely talked about as Web 2.0 items. Web 2.0 is an imprecise term but it usually means sites that have a high degree of interaction with visitors including some features visitors can personalize.

Often, the coding technology used to build Web 2.0 sites is not garden variety HTML but rather a beefed up type of coding scheme. One group of such schemes is called AJAX, which stands for Asynchronous JavaScript and XML. AJAX is not a single technology but rather it is any of several web page presentation schemes that transform the various object models that are built into browsers (no two of which are exactly the same) into a uniform page model that uses libraries of JavaScript to provide functionality that goes far beyond the basics.

AJAX pages that are in wide use generally have a contemporary look and feel. These days what seems popular is to fill a page with items that have rounded corners and gratuitous shading. For many visitors AJAX layouts make you wish you were sitting at a screen that's twice the size of the one you have no matter what size screen you have. Still, despite their quirks, today's AJAX webs are very popular on sites aimed at trendy visitors.

Tomorrow's popular web styles may look nothing like the sites today seem to be at the leading edge. That possibility of change is one reason AJAX is used for webs that must follow design trends. Changes made once in AJAX designs turn into changes for every version of every page. By contrast, webs that are coded by hand to match browser quirks may require many updates to implement a single change in appearance. This isn't a big issue on web sites that don't push presentation technology to the limits because all web browsers can handle common HTML and CSS directives, but sites that strive for unusual effects tend to use technology that is not presented uniformly by every browser and these are the sites where AJAX has appeal.

Lots of less obviously contemporary pages on corporate web sites use AJAX, too, but do so in a way that makes the technology seem more ordinary, gentle and inviting to the casual (and generally speaking more traditional) visitor. Some of these corporate sites suffer from the excesses that plague consumer AJAX sites, like the gratuitous use of moving images that are more

distracting than informative, but more often than not the mix of Web 2.0 ideas and AJAX technology just makes these webs easier to navigate and their web pages easier to read.

A quality web experience can give any business a boost, and now more than ever businesses want every extra bit of help they can get. Harsh conditions are forcing many companies to rethink how they use their webs and also how they build and maintain their pages. For some, GWT and Eclipse might be one route to a more effective web presence.

The technology in the Google Web Toolkit originated within Google or inside firms Google acquired as it built up its web development capabilities. Google didn't develop and refine this technology for the general public, not at first. GWT is what Google used to build the visible parts of its web applications such as Gmail, Google Docs and other elements of its cloud computing service. Google developed its technology in a way that makes the client machine do a lot of the heavy lifting, thus helping Google serve more visitors to its cloud systems with less server power. That's a big issue when you work in front of an audience of millions.

Google makes its concept pretty clear when it explains how its toolkit works: Developers build elements of a web page (or a whole page) with Java. Then they test it as Java, which, compared to testing in JavaScript, makes it easy for software to identify many mistakes. This is because Java is a tightly specified language with strongly typed data elements and a strict grammar. By comparison JavaScript, which may start with the same four letters but is otherwise quite distinct, is made for scripting and for many reasons is tolerant compared to Java. This can be useful for script writers but it's murder when it comes to debugging. A lot of JavaScript software is tiny, so even when it has an error the programmer can spot the problem with a quick inspection. But that is not the case with AJAX systems. The JavaScript is big and complicated and there are no debugging tools that work as well as the ones associated with Java, not because programmers are indifferent but because JavaScript, once it gets bulky, is a nightmare to analyze.

Anyway, once a developer likes the way an element or web page behaves with its GWT Java, Google's software compiles the Java into JavaScript. The compiler seems to be pretty good. If the Java works so will the JavaScript. There are blogs and newsgroups where developers talk about their problems and the level of user satisfaction with GWT, at least as far as the compilation accuracy is concerned, seems very high.

In practice, GWT doesn't make one JavaScript version of a web element or web page, it makes many, one for each kind of browser or more accurately one for each browser with a distinct JavaScript interpreter. GWT also provides a scheme for browser detection and so when a visitor hits a GWT web page the client system gets the right package of JavaScript. This process can still make web pages pretty big compared to hand coded HTML pages, but GWT output is skinny compared to the output of AJAX systems that move a ton of code to the client and then let the client select the section of code that fit its JavaScript interpreter and document model.

The GWT system may at first look like it favors Linux and Windows but that's an false impression. There is a Java runtime environment for just about every operating system you can think of, and a Java runtime is about all a system needs to support GWT. That means developers can work on a PC, on an i5, on a Linux box, on a mainframe, on a Unix system . . . you name it.

When the web pages are all compiled the resultant JavaScript can be served by pretty much any web server, but Google's apparent focus is on the two most popular web servers, Apache and Microsoft's IIS. Because Apache is available for every IBM platform (and can coexist with IBM's Websphere applications server) developers with servers from Big Blue should not have any trouble building pages with the Google Web Toolkit and hosting them on whatever system they chose because, very likely, they prefer the way it does bookkeeping or the value it provides for a favored applications suite. Once the Google object pages with their JavaScript AJAX are on a web server they seem to speak correctly to pretty much every current browser. That means not only Internet Explorer or Firefox but also Safari in an iPhone and Konqueror on a high-powered Unix workstation. Google seems to have achieved its intention. When the GWT is used correctly, all servers and all browsers become equal, or, more importantly, they are made to look and feel that way.

Now it turns out that Java developers don't just write their code using a text editor, even though they could. Generally speaking, Java coders work with a set of programming tools called an Integrated Development Environment or IDE. And it turns out that just as Java is part of the Open world, so, too, is a very good IDE called Eclipse that is well supported by IBM. Eclipse began life about ten years ago at an IBM Canada subsidiary first called Object Technology International but now known as IBM's Ottawa Lab. Three years later in 2001 IBM moved the Eclipse project to a group called the Eclipse consortium and made its code public in an effort to rally the Open Source crowd around it. Unfortunately, the Open community still felt the heavy hand of IBM on Eclipse and

basically didn't pitch in to improve and popularize the software. In 2004, IBM, which wanted Eclipse to become a de facto standard for Java development, set up an independent nonprofit organization called the Eclipse Foundation, moved the software into it and made it clear that Big Blue would let the Foundation make its own choices from then on. This was the right move as far as the politics of Open software were concerned.

So, it's taken a while, but Eclipse has gained a following. Today it not only provides an IDE for Java but also helps developers working in other languages, such as C and C++. There are IBM is still a big supporter, but IBM's rivals such as Oracle are behind Eclipse, too, now that it is seen to be guided more by its community than its sponsors. The Eclipse community is quite lively and it likes to share inventions that extend and improve the core Eclipse environment.

Today there are quite a few plug-ins that extend Eclipse, some free, some commercial; there is even a web site for Eclipse add-ons. There are also companies that, for a fee, provide and support software distributions based on Eclipse, much the way there are outfits like Red Hat that sell Linux distributions and Linux support.

When it comes to influencing web developers, all these players together might not have the clout of Google, so Elipse fans are pleased that Google has embraced the IDE they prefer. Google's support is concrete, and it comes in the form of advocacy as well as software. Sure, Google Web Toolkit works fine on its own. But Google also has put facilities in its Toolkit to help the Google software integrate seamlessly with Eclipse. Using GWT with Eclipse generally just means telling the Google Toolkit your intentions when you first set up a project and then telling Eclipse to import the embryonic project created by GWT into what it called the Eclipse workspace. All the details are of importation are automated and the two software packages appear to cooperate as promised. The result is an Eclipse development project that creates and tests web pages running Java and which then uses the GWT to turn the Java web page elements into client side components written in JavaScript.

Just to help you get started, here are URI's of some resources:

Google Web Toolkit Home: http://code.google.com/webtoolkit/

Getting Started with GWT: http://code.google.com/webtoolkit/gettingstarted.html

GWT Blog: http://googlewebtoolkit.blogspot.com/

GWT Official Group: http://groups.google.com/group/Google-Web-Toolkit

GWT Unofficial Group: http://www.gwtsite.com/

Eclipse Org Web: http://www.eclipse.org/

IBM Developerworks Eclipse Section:
http://www.ibm.com/developerworks/opensource/top-projects/eclipse-starthere.html

Eclipse Plugin Central: http://www.eclipseplugincentral.com/

— Hesh Wiener January 2009