Fun Factor: Coding With XQuery

A Conversation with Jason Hunter

by Ivan Pedruzzi, Senior Product Architect for Stylus Studio

Jason Hunter is the author of Java Servlet Programming and co-author of Java Enterprise Best Practices (both O'Reilly Media), an original contributor to Apache Tomcat, and a member of the expert groups responsible for Servlet, JSP, JAXP, and XQJ (XQuery API for Java) development. Jason is an Apache Member, and as Apache's representative to the Java Community Process Executive Committee he established a landmark agreement for open source Java. He co-created the open source JDOM library to enable optimized Java and XML integration. More recently, Jason's work has focused on XQuery technologies. In addition to helping on XQJ, he co-created BumbleBee, an XQuery test harness, and started XQuery.com, a popular XQuery development portal. Jason presently works as a Senior Engineer with Mark Logic, maker of Content Interaction Server, an XQuery-enabled database for content.

Jason is interviewed by Ivan Pedruzzi, Senior Product Architect for Stylus Studio. Stylus Studio is the leading XML IDE for XML data integration, featuring advanced support for XQuery development, including XQuery editing, mapping, debugging and performance profiling.

Ivan Pedruzzi: Hi, Jason. Thanks for taking the time to talk with *The Stylus Scoop* today. Most of our readers are familiar with your past work on Java Servlets; but could you tell us what was behind your more recent interest and work in XQuery technologies?

Jason Hunter: Hi, Ivan – I started using XQuery back in the summer of 2002. What struck me immediately was that XQuery coding was fun — it had an appeal reminiscent of Servlet and Java programming. By contrast a lot of the other XML work I've been doing with Schema and DOM are, well, a lot less fun. If a particular technology is important-yet-tedious, it may succeed but won't blossom because people won't feel any allegiance to it. In this respect, XQuery is happily more like servlets than like Schema or DOM and I find that to bode well for its future. But obviously, a successful technology can't be just fun, it has to be useful. The usefulness of XQuery comes from its ability to query nearly any kind of content: XML documents, relational stores, PDF files, or Microsoft Office documents. XQuery seamlessly integrates and binds them *all* together.

So for example, I used to write websites in Java; these days I'm doing more with XQuery. You can integrate XQuery behind a J2EE server (I have a JSP tag library for this) but I've found it easier to simply put XQuery directly on the web. What you do is put .xqy files on the server that are XQuery scripts to produce XHTML pages. It works like a CGI or JSP, but instead of running Perl or Java, the server runs XQuery. XQuery has native support for XML and XHTML, so writing XQuery pages is an amazingly easy way to put back-end data on the web. There's no impedance mismatch. In just 250 lines I wrote a web-based RSS reader with cross-feed search. With J2EE, that would be the size of just the deployment descriptor!

Of course a new technology must be practical, too. There are just too many new technologies out there hoping to solve really abstract problems that are too obscure to actually be useful in the real world. XQuery is practical in a number of ways as it provides a very elegant solution to some of the most commonly occurring problems in enterprise computing. As a consultant, the best piece of advice I ever received on prospecting potential clients was to "*find the person with the money – figure out what keeps him or her up at night and solve that problem.*" There are certainly many enterprise customers working to try to access multiple data and document sources – XQuery enables

you to access content from them all in a simple yet powerful way. Overall, when you consider XQuery's usefulness, practicality and what I call its 'fun-factor', it's quite a compelling technology, and definitely worth looking into, in my humble opinion.

IP: Tell us about the work you are doing as a member of the Expert Group behind the XQuery API for Java (XQJ) Java Specification — can you explain for our users what XQJ is?

JH: XQJ is the XQuery API for Java, an effort to create a standard mechanism to execute XQuery from Java. What JDBC is for SQL, XQJ is for XQuery. It's still in early drafts, but I do expect it to be quite an important Java specification. Right now, XQuery vendors each have their own server access APIs. That's not so bad because the APIs can be learned in an afternoon and ported between in an hour, but it's better to have real portability and a standard access mechanism. The individual implementation ideas are feeding into the XQJ designs.

There's a bit of irony in how the expert group is run. As the Apache Software Foundation's representative to the Java Community Process Executive Committee, I fought for more open licensing of Java specifications and more transparent and cooperative operation of expert groups. XQJ is among the first groups to take these ideas to heart. We're publishing early and often under liberal licensing terms, and we're collaborating on the specification like an open source group. That means if you point out a problem, you fix it and send in a specification "patch". As my patch to-do list has grown and grown, I've wondered if the old spec-lead-does-everything way wasn't better!

IP: Is XQJ as important as say, JDBC or ADO.NET is in terms of simplifying programmatic database access?

JH: Standards like XQJ encourage diversity above the standard. This is to say that without a standard access mechanism like XQJ, it would be difficult for developer tools and programs using XQuery to keep up with so many different vendor data access technologies. For example, with BumbleBee, an XQuery test harness I've been involved with – we had to write separate adapters to each vendor's access API. Having a standard would let us focus on adding more advanced features rather than compatibility with each XQuery vendor.

The design of XQJ is actually based on JDBC and the expert groups talk regularly about working effectively together. A common question is, will XQuery be the next SQL? My prediction is that XQuery and XQJ won't replace SQL and JDBC, but will take away the spotlight. Jim Melton from Oracle (and co-lead of XQJ) recently opined in his corporate web log, "*Like it or not, the SQL standard is in its twilight years, with XQuery poised to overtake it in terms of major new applications by 2010.*"

IP: Do you think that the advent of higher level XQuery and XQJ technologies signals the end of JDOM programming or other low-level manipulation of XML trees? (*Editor's note: Jason was the principle developer of behind the creation of the JDOM Java XML modeling libraries*)

JH: Ha! Certainly there are tasks where I once used JDOM where these days I would much rather use XQuery instead, like processing multiple RSS feeds. But JDOM is alive and kicking. In fact we just released the long-awaited JDOM 1.0 this month!

JDOM still has (and in all likelihood will continue to have) an important place in the XML application stack. XQuery uses XML as a native data type, and that works well as long as you're operating solely within an XQuery environment. As soon as XQuery has to export or 'bind' data to some programming language like Java, C# or whatever, the XML needs to be exposed using some object representation. If you're programming in Java, then I happen to think JDOM is the perfect XML-to-Java or Java-to-XML 'glue'. For example, in XQJ you can plug in a content handler to return JDOM node fragments. Ultimately the greatness of XQuery stems less from replacing JDOM functionality than it is does from enabling far more advanced functionality that was previously either not possible or really difficult to do, such as querying and integrating distributed data sources. Yes, the use of XQuery typically pushes JDOM code a level deeper in the overall application stack but given the added power and extensibility of XQuery I can't see this as anything other then a positive development.

IP: Are there certain kinds of applications that are naturally better suited for use with XQuery?

JH: Obviously there are many uses of XQuery but I find that one of the most interesting uses for XQuery is in managing content. The world has a lot of document content that doesn't fit so neatly into the rectangular boxes imposed by relational databases. Examples are medical records, textbook materials, office documents, and web pages. You can store these documents directly into an XQuery database – this way, you can avoid the previously obligatory "shredding" (decomposition) to and from relational database format, and instead directly

query the documents to extract the specific bits and pieces deemed important.

Document and content management systems of the past concentrate on content development (and the workflow activities around that). XQuery now offers the ability to do something interesting with the documents besides just publishing them as-is. You can take your documents as input, along with other documents on your desktop or downloaded from external sources, and store them into a content database. Then you can drill into the documents to find, identify, extract, and render the most relevant pieces. With XQuery, the document is the database.

I joined Mark Logic this January because they're doing awesome work in this area. One Mark Logic customer, Elsevier, has stored more than a terabyte of reference books, journal articles, research reports, and directories into the Mark Logic engine. Using XQuery and the Mark Logic fast text search extensions, Elsevier can search against any or all of the data set and return custom search results. Normal search engines return pointers to answers: "We found the phrase 'tennis elbow' in these 23 textbooks and 15 journal articles." XQuery, by letting you manipulate XML nodes and walk up and down the XML tree, lets you generate advanced search reports whose content is under the control of the customer, for example: "Here are the top 10 most relevant paragraphs on 'tennis elbow' displayed in context as part of the nearest containing subsection. "

Additionally, XQuery makes everything more easily customizable. In one situation, I had to use XQuery to generate a "press view" of medical journal articles. I simply wrote an XQuery that extracts all the images and tables (the "eye candy") and, in order to provide context, also extracted the first paragraph that referred to each media item. I also took the appropriate blurb from the journal issue's table of contents and popped it at the beginning as a high-level overview. I could have quite easily have further customized the content and that's what's so fun about XQuery.

IP: Well we've certainly heard great a lot of great things from the Stylus Studio developer community about the XQuery support in Mark Logic Content Interaction Server. They have quite the reputation for scalability and performance. Do you have a demo of these applications?

JH: Yes. We created an online demo application — it uses thousands of public SEC filings which list information pertaining to the salary, bonus, and stock options of top executives at publicly traded companies, along with corporate overviews and competitor sheets from Hoover's. The application lets you compare and contrast executive compensation. All the content comes in HTML, gets normalized and extracted by XQuery, and is searched and displayed with XQuery. It's a great example of how you can work with structured data along with both semi-structured and unstructured content using XQuery.

IP: How do you go about building these advanced XQuery applications?

JH: The availability of XQuery development tools is an important factor in getting folks to adopt these new technologies. A huge benefit of Stylus Studio is that it helps make advanced XQuery easier for newcomers to the technology, and hence more accessible, which is really important for me as much of my work involves training people on the use of XQuery and other new technologies. So for example, I often use Stylus Studio in my XML training classes to demonstrate XQuery concepts and the students definitely get a better understanding right away. I'm impressed with the overall integration of Stylus Studio's XQuery-aware editing, mapping, debugging, and performance profiling tools — it's quite a productive XQuery development environment.

One of the key challenges I see for Stylus Studio going forward will be adding support for other 3rd party XQuery engines in a similar fashion to how Stylus Studio's XSL/XSLT tools currently work with all the XSL/XSLT processors out there. Stylus Studio's XQuery profiler, for example, could then recognize that executing //foo[bar="quux"] across a Gigabyte can execute in milliseconds or minutes depending on indexing, and has to talk to the production server to know its capabilities and settings. It also would be great if the debugger and editor understood vendor-specific capabilities, like Mark Logic's fast text search extensions, updates, try/catch blocks, blob support, etc.

IP: I tend to agree — we should probably collaborate on that ... have your people call my people?

JH: Sounds like a plan! [laughs]

IP: Where can developers learn more about XQuery?

JH: For general XQuery information, I recommend xquery.com. It's a watering hole for XQuery developers with a mailing list where people can ask questions and support each other in learning a new language. It's vendor neutral and has plenty of helpful XQuery resources.

To get started programming XQuery, I encourage people to download the free, non-expiring Community License to Mark Logic Content Interaction Server. It's fully functional, just limited to 50 Megabytes of content. You can also get a 30-day Trial License that holds up to a Gigabyte. Likewise, I also encourage people to get the free 30-day evaluation copy of Stylus Studio. You guys have some great tools for working with XQuery.

IP: Hey, thanks!

JH: Well, I mean it. It's a great tool for beginners exploring XML technologies, and for advanced users who are ready to exploit it. I'm excited to see what the future brings to Stylus Studio as you guys continue your progress.

IP: Thanks again for your time, and for your kind words.

xq:zone
where xquery gets to work

Mark Logic Corporation
2000 Alameda de las Pulgas
Suite 100
San Mateo, CA 94403
+1 650 655 2300 Phone
+1 650 655 2310 Fax
www.marklogic.com
xqzone.marklogic.com

Stylus Studio
14 Oak Park
Bedford, MA, 01730
+1 781 280 4488 Phone
+1 781 280 4295 Fax
www.stylusstudio.com
stylusstudio@stylusstudio.com