Vol. 11 No. 4, July 2006

Watch this: Ajax, YUI and Web 2.0

Terrence A.Brooks
The Information School
University of Washington
Seattle, WA 98195

Can't even understand the title of this column? Jargon check first!

- **Ajax**: 'Asynchronous JavaScript and XML'. Your Web browser uses a scripting language (i.e., JavaScript) to connect to a Web server and ask for more information. This information may be received a moment or two later (i.e., asynchronously) and be packaged as Extensible Markup Language (i.e., XML). For some applications, unpacking XML may be too laborious and a more light-weight information structure, JSON (i.e., JavaScript Simple Object Notation), is used. What does this mean to you? *Web pages offer richer information experiences as only portions—not the whole page—are updated.*
- **YUI**: 'Yahoo! User Interface Library'. A JavaScript library that hides the ugly details of cross-browser scripting and permits anyone to build a Web page that uses Ajax and sophisticated user-interaction widgets. If you have a sense of Web-technology history then you recognize that this is (*finally!*) the arrival of DHTML (Dynamic Hypertext Markup Language). What does this mean to you? *Web pages will feature the look and feel heretofore only seen on the desktop (e.g., slider widgets, calendar widgets, combo button widgets, etc.)*
- **Web 2.0**: Jargon that signals that the static Web pages of the first decade of the Web are passé. Alternatively, it is the new name of DHTML.

Gist of this column: Web pages gain functionality to rival desktop applications (Web 2.0 Awards)

A little history first: Today's Web browsers trace their lineage back to the 1990s browser wars when major Web players attempted to dominate the Web by building browsers with idiosyncratic implementations of JavaScript, Cascading Style Sheets, the Document Object Model, etc. One unintended consequence was the increase in complexity of cross-browser scripting. It was tough to build a fancy Web page when different browsers (and different versions of the same browser) broke your Web page. *Interesting background reading:* "Internet Explorer vs. the standards".

As we leave the age of DHTML behind us, cast a glance back at perhaps the last and best DHTML book: *DHTML utopia: modern Web design using JavaScript & DOM* by Stuart Langridge. Note how Langridge has to develop hacks to deal with cross-browser issues such as memory leaks and closures. Most people who create Web content don't want to tangle with these subtleties. Hence the utility of JavaScript libraries (i.e., YUI) written by experts that hide all the ugly cross-browser details so the rest of us can concentrate on our Web content.

Conceptual fork: Note that there are two topics here. One is making a server call for new content. Then with the new content in hand, you need to reach down into the HTML structure of your Web page (i.e., its DOM - Document Object Model) and add/change/delete/modify, etc. Ajax libraries cover both of these. The first is relatively simple, but the second is much more messy. Can you imagine writing JavaScript code to create a slider object that works on PCs and Macs, Internet Explorer, Firefox, Opera, etc. *Tricky!*

Ajax - Asynchronous JavaScript and XML

February 18, 2005 Jesse James Garrett wrote *Ajax: a new approach to Web applications* and introduced the paradigm of a Web page making many small requests back to its Web server. The classic pioneering Ajax application was Google Maps, which seemingly presented a huge map that you could view through a draggable viewing port. It really wasn't a huge map, but many small maps that were downloaded as you dragged the viewing port around.

Writing your own Ajax JavaScript procedure is surprisingly easy. The first three hacks in *Ajax hacks*" by Bruce Perry are (1) Detect the what type of client browser is going to run your script, (2) Create an XMLHttpRequest object with the URL of the server of origin, and (3) create a call-back procedure to handle the information package from the server. By page 17 of his book, you've written your 30 or 40 lines of JavaScript code and, suddenly, you're the master of the XMLHttpRequest object.

Creating your own Google maps application turns out to be easy as well. *Pragmatic Ajax: a Web 2.0 primer*" by Justin Gehtland, Ben Galbraith and Dion Almaer walks you through the code in one chapter.

The XMLHttpRequest object from the point of view of Web readers and writers:

Hypertext Transfer Protocol (HTTP) is the standard protocol for exchanging information between client and server and when combined with XML is referred to as XMLHttpRequest object. By 2002 most browsers supported the XMLHttpRequest object.

Web writers engineer Web pages to respond to user events by reaching back through XMLHttpRequest to the *server of origin* and asking for updating information. JavaScript security limits XMLHttpRequests to the same server that gave you the Web page. Without this safeguard, a Web page could collect your information and send it to an evil third party.

Web readers, on the other hand, use the XMLHttpRequest object in Greasemonkey to mash various Web pages together and can reach out to as many servers as they like. Authoring your own mashups means no evil third party is going to get your information.

Don't forget!: It's easy to say that a Web browser will make a request to a server. This implies, however, that the target server has programs ready to receive these requests and respond appropriately. The complement of your client browser Ajax script is a server script written in CGI, PHP, ASPX, etc.

Did you know that 'ajaxian' is an adjective

The neologism 'ajaxian' describes a Web page experience beyond the static Web pages of the first decade of the Web. Del.icio.us has a tag 'ajaxian', there is an ajaxian Website, and there are calls to conferences on the Ajax experience. All this heralds the arrival of JavaScript toolkits that give us easy-to-deploy widgets that work regardless of the browser.

My favorite is YUI: Yahoo! User Interface Library. I suggest the strategy of merely expropriating their examples, modify the example's contents and style and, *bingo!* you have a sophisticated widget that works cross-browser. YUI has documentation and even cheat-sheet summaries of widget capabilities. Best of all, there seems to be a sense of corporate responsibility in supporting the community of users of YUI. The e-mail I received this morning is an example:

> Due to issues surfaced by the community after the 0.11 release last week, we've issued an update today and we recommend that everyone using 0.11 update to 0.11.1 (available on SourceForge). This update resolves critical issues in the Animation Utility, the Dom Collection, and the Container Control. A manifest of changes is available on SourceForge and the full changelog for each component can be found in its README file.
>
> Meanwhile, we continue to work on the popular YUI Cheat Sheets. These were updated for all the Utilities when we released 0.11 last week, and we added Cheat Sheets for the Logger, Slider and TreeView controls at that time as well. Along with today's update we've added a Cheat Sheet for the Calendar Control.
>
> Regards,
> Eric

For the habitués of the leading/bleeding edge this e-mail signals *somebody cares!*

The other compelling reason promoting YUI use is Yahoo's Design Pattern Library. Here are formats and strategies for the common interaction problems of Web pages. In short, Yahoo is creating Web 2.0 by offering us both the tools and the strategies for their use.

Some Ajax libraries

- YUI: Yahoo! User Interface Library has the advantage of a corporate sponsor that wants to develop a user community, solid documentation, cheat-sheets and an active development program. *My first widget worked in minutes*.
- Dojo is an active, open-source JavaScript project that is building a comprehensive set of JavaScript procedures. Unfortunately it lacks adequate documentation. It reminds me of a group of developers, all of whom are so busy developing new code that no one has written a ramp-up for the novice. There are nightly builds, but the non-developer would have difficulty incorporating these into a Web page. *Damn! Two weeks of frustration and I still can't get the combo button to work! Why isn't there a simple example to follow?*
- Prototype is heavily influenced by the Ruby on Rails framework, which focuses on the Apache server. Rico and Script.aculo.us build on Prototype and offer relatively small libraries of widgets
- Google Web Toolkit suggests that, to create a JavaScript, we should write some Java code and compile it. *What were they thinking?*

- Atlas is for Microsoft's ASP.Net community.

Date: July 2006

For further reading:

[Ranked according to my taste]

- **Ajax hacks** by Bruce W. Perry. O'Reilly, 2006. Excellent! Gives the code, and then gets out of the way. Examples include Prototype with Rico, Ruby on Rails and script.aculo.us libraries. Doesn't cover Ajaxian widgets. The first chapter is must reading.
- **Build your own Ajax Web applications** by Matthew Eernisse. Sitepoint, 2006. Very good for folks who

want to be talked through the code. Builds a basic Ajax application and covers Web services and some Ajaxian widgets. Perhaps the best book for the intelligent layman.

- **Pragmatic Ajax**: A Web 2.0 Primer by Justin Gehtland, Ben Galbraith and Dion Almaer. The Pragmatic Bookshelf, 2006. Good as an overview of many initiatives in the development community.
  Note that one of the authors is a prime mover of dojo. Quote: "While we're on the subject of Dojo, have we mentioned how much we like it? We think Dojo, of all the existing JavaScript toolkits, has the most potential to break away from the pack and become a dominant player in the space. In fact, we feel so good about Dojo that we believe it will become obviously more popular than any other Ajax framework" (p. 284). *I respectfully disagree with this blurb.*
- **Head rush Ajax** by Brett McLaughlin. O'Reilly, 2006, Written in the famous 'head rush' style. *Cool! Righteous! Rad!*, but if you're interested in studying code then the comic book style can get in the way.