

■ KENNETH E. KENDALL, Feature Editor, School of Business-Camden, Rutgers University

In this month's column, Allen Schmidt and I explore a technique that uses a collection of technologies dubbed "Ajax," that controls how information is displayed on a Web page. All e-commerce consumers and Web page developers can gain competitive advantages from knowing a little bit about Ajax. It has many benefits in Web site programming and human computer interaction including, but not limited to, efficient loading of information to the less cluttered design of a Web page. Most of the Web features that Ajax delivers are laudable but, as with other technologies, there may be some features that are not in your best interest. Read on. [Kenneth E. Kendall, Feature Editor]

Using Ajax to Clean up a Web Site: A New Programming Technique for Web Site Development

by Allen Schmidt, Madison Area Technical College;
Kenneth E. Kendall, Rutgers University



Allen Schmidt

teaches JavaScript and Systems Analysis and Design in the Information Technology Department at Madison Area Technical College (MATC) in Madison, Wisconsin. He has written four books on CASE tools and has

co-authored an interactive, Web-based simulation that teaches systems analysis and design called HyperCase®.

schmidt@chorus.net



Kenneth E. Kendall

is president-elect and a Fellow of DSI. He is a professor of e-commerce and information technology at Rutgers University, School of Business-Camden. He is also a co-author of HyperCase® which can be

found at <http://www.thekendalls.org/hypercas.html> and co-author of Systems Analysis and Design, 6th edition.

ken@thekendalls.org

When getting ready to travel between countries in South America, we noticed that it was not possible to use the Web to pull up our flight between Santiago, Chile and Buenos Aries, Argentina. Although we possessed an electronic ticket, we wanted to see whether the departure time had changed. That option was not available to someone using the Web site from an IP address in the US. When we arrived in Santiago, we went to the airline's home page and there was an additional box displayed on the page. This allowed us to view the flights between Chile and Argentina. Something on the Web site was able to identify where we were accessing the pages from. Ajax was used for this purpose (and also to provide an option to read the page in Spanish or English). It was not available to users who were located in the US using an IP address from North America, but Ajax made it possible for someone using a computer in South America to receive this information.

What is Ajax?

Ajax is not a new programming language, but a technique that uses a col-

lection of existing technologies to improve the functionality of a Web interface to give it more of the look and feel of a desktop application. Rather than making users wait for a response from a Web server, Ajax delivers just enough information to keep the interaction going in a human-paced way. Ajax works with the later browsers, such as Internet Explorer 6.0, Firefox, Netscape 6.0 and newer versions of these browsers.

Ajax uses JavaScript, a scripting language that enables Web authors to design interactive sites (*Webopedia* [2006a]) and either XML, which allows designers to create their own customized tags that enable the definition, transmission, validations, and interpretation of data between applications and between organizations (*Webopedia* [2006b]); or plain text to modify portions of a Web page in response to a user action. In the traditional Web environment, the viewer may make a selection from a drop-down list, and the Web page responds by sending a request to the Web server. The Web server sends a new Web page back to the browser based on the choice made in the selection list.

By contrast, an Ajax enabled Web page would respond to the same change

in a drop-down list by sending a small amount of text to the server, which would return either plain text or an XML document to the same page. The text or XML is used to format the next portion of the Web page. Your experience as a user is smoother since you do not have to wait for a new page to display, and you do not have to figure out where to continue on the next page.

When Should Web Developers Use Ajax?

Ajax can be used when a Web form with a drop-down list or radio button or other control is used to obtain additional data for a transaction from the Web server. Developers often need to partition Web sites into a series of pages at critical points, such as when a secure Web page must be loaded or to avoid information overload for the user. When a developer creates a page to assist a user in completing an airline reservation, the Web site might be partitioned into manageable, logical portions for the user including: finding a flight, entering or selecting the number and names of passengers, paying for the flight, selecting a seat, and selecting a specialty meal, or other special needs.

Ajax may also be used to obtain an XML document from a Web server and then use the data to display information on a Web page. The advantage of using XML is that you may have many of what would have been records in a traditional system stored within one XML document. The records may be selected from within the XML document to display information about them.

One example is displaying information for selected customers. The Web browser would use a form for users to enter a zip or other mailing code, telephone area code, or any other selection criteria. This would be sent to the server which would obtain the selected records from a database table or tables. The results would be formatted as an XML document and sent to the same Web page. The Ajax code coupled with dynamic HTML (hypertext markup language) would allow the selected cus-

tomers information to be displayed, with buttons to scroll forward or backward through the customers, one at a time. This is an easy to use interface, and eliminates the wait time for loading new Web pages. It also allows customer information to display one customer at a time, eliminating scrolling through a lengthy page of irrelevant information.

Examples of Ajax

A good example of Web pages using Ajax is *Google Suggest*, which you can find at <http://www.google.com/webhp?complete=1&hl=en>. When the viewer starts keying in the letters for **Decision Sciences**, Google will update a drop-down list of search topics with the number of results. Google is using Ajax in many different applications and users are responding enthusiastically.

Ajax techniques are used to modify a Web page, increase response time, cut down on user waiting, and provide a smoother user experience with e-commerce pages and other interactive Web sites.

Another example is *Direct Ferries*, located at <http://www.directferries.co.uk/poirishsea.htm>, which uses the Web to allow viewers to make ferry reservations. On the home page, you can select an outward and return route, as well as the number of people traveling. On the second page, you can choose the type of vehicle you want for land transportation. Notice that the Web page expands with additional drop-down lists. Choose a make of auto desired, and the model drop-down list is populated. The Web page is modifying itself using dynamic HTML code. This is a change since spring of 2006, when this site loaded a unique Web page for each choice that was made.

Changes to Website Design

To make Ajax effective and avoid user confusion, the Web page designer must include clear instructions, particularly

if there is a concern that the viewer may not notice that the Web page has changed.

Since the Web page is dynamically changing, feedback must be provided to inform the user about what to expect. An example would be a drop-down list initially containing one option stating "Selection Currently Not Available." When a message is sent to the server to obtain the values and text used to create the drop-down list (based on selections elsewhere on the Web form), the first option would change to something like: "Please Wait – List Is Being Populated." When the server returns the text or XML used to populate the list, the "Please Wait" message is removed.

How Ajax Works

The Web page sends a request, typically containing a small amount of text, to the server. A number is returned indicating the status of the request, starting with 1 and ending with 4. When the status number reaches 4, the data has been returned and the "Please Wait" message is removed using dynamic HTML. This is done using the document object model or DOM. Every object may be given an ID, which allows the JavaScript code to store a reference to the object in a variable. Once this reference is obtained, JavaScript may be used to modify the object or any of its attributes.

In the example of changing an option in a drop down list, a reference to the list is stored in a variable. Each option in a list is considered a child of the list, and JavaScript uses a command called "removeChild" to delete the current text (originally stating "Selection Currently Not Available"). Next, JavaScript uses a command called "createElement" to create the new text ("Please Wait – List Is Being Populated") and another command, "appendChild," to add the text to the drop-down list. When the text used to populate the list with actual values is received from the server, the "removeChild" command clears the "Please Wait – List Is Being Populated" message and JavaScript uses a loop to "appendChild" to create the list with the new values from the server.

Lists may be changed by using simple text received from the server. More complicated activities, such as displaying customers and allowing the user to view them one at a time, involve the use of XML. When an XML document is received from the server, the elements in the document are stored in an array or matrix. As the user moves through the different customers (or other XML data), the JavaScript code gets the XML data from the next entry in the array and formats the Web page. The XML data may also be used to populate a drop-down list of customer names, and when the user changes to a new name, the Web page displays the matching customer.

These are just a few of the Ajax techniques that may be used to modify a Web page, increase response time, cut down on user waiting, and provide a smoother user experience with e-commerce pages and other interactive Web sites.

Ajax has Drawbacks, Too

One of the drawbacks of using Ajax is that the *Back* button on the browser will not provide previous page information that existed before the selections were made. This is a serious usability issue and it violates widely embraced user expectations and customs. There must be a way for the Web page to allow viewers to change information that led to the current selections.

Another drawback is that Ajax relies on JavaScript, and about twenty-five percent of viewers have JavaScript disabled on their computers.

Ajax Implications

Web developers can reduce the wait time a user typically experiences when interacting with an e-commerce or other Web site. Ajax keeps the conversation with the user fluid, and makes it smoother. Ajax makes loading of Web pages more efficient. Information can be displayed rapidly. Ajax can also improve how a screen looks. Data that are not pertinent to the current context can

be hidden from the user so that the Web page has a cleaner, uncluttered design.

Like all technologies, a feature can turn out to be detrimental if not used properly. Ajax might be used to censor information and, in effect, bar you from seeing it. A corporation may (in conjunction with the Web site) use Ajax to prevent information from displaying on your screen. One possibility is that the corporation is protecting you. Or maybe it is trying to prevent you from doing something (like allowing a U.S. citizen to purchase an airline ticket at the same price available in South America.) Alternatively, it is possible that a corporation might use Ajax to receive only the information it wants you to see on a given Web page.

References

Direct Ferries. (2006). *www.directferries.co.uk/poirishsea.htm*, last accessed September 16, 2006.

Google Suggest. (2006). *www.google.com/webhp?complete=1&hl=en* last accessed September 16, 2006.

Nielsen, J., Molich, R., Snyder, C., & Farrell, S. (2001). *E-Commerce user experience*. Fremont, CA: Nielsen Norman Group.

Webopedia. (2006a). Definition of JavaScript, *www.webopedia.com* last accessed on September 23, 2006.

Webopedia. (2006b). Definition of XMLt, *www.webopedia.com* last accessed on September 23, 2006. ■

2006 DSI Annual Meeting Website Links

DSI Annual Meeting Homepage
www.dsi-2006.org

Online Conference Registration:
www.decisionsciences.org/CIS

Hotel Reservations
www.stayatmarriott.com/DSI2006/

2007 SEDSI Student Paper Competition

Annual Meeting to be held February 21-23, 2007, in Savannah, GA

The student paper competition is designed for students currently enrolled in PhD programs. Please refer to the www.sedsi.org website for submission procedures. Student papers are subject to the same review process as regular papers. All student papers accepted by the reviewers will be scheduled on the program and a \$100 travel stipend will be awarded to each paper (regardless of the number of authors) at the meeting. Students who do not present their papers will not be eligible for awards. The Student Paper judges will also evaluate the quality of the presentations and recognize outstanding papers at the meeting luncheon on Friday, February 23rd, 2007. **Submission deadline is November 20, 2006.**

DSINFO

DSINFO, a listproc maintained by the Decision Sciences Institute, broadcasts emails on news and announcements relating to DSI and the decision sciences community. The listproc can be used for announcing calls for papers and for updating news on meeting and other events.

DSINFO subscribers also receive notice from DSI when *Decision Line* articles and *Decision Sciences* abstracts are made available on the DSI website. Because this content is placed on the website prior to printing the hard-copy, the articles/abstracts are available on the Internet weeks before the publications arrive in the mail.

For more information on joining DSINFO or to subscribe, visit

[http://mailbox.gsu.edu/mailman/listinfo/dsinfo](mailto:mailto://mailbox.gsu.edu/mailman/listinfo/dsinfo) ■