

# JAVASCRIPT & AJAX ACCESSIBILITY

Becky Gibson  
Web Accessibility Architect  
IBM Emerging Technologies  
5 Technology Park Drive  
Westford, MA 01886  
Email: [gibsonb@us.ibm.com](mailto:gibsonb@us.ibm.com)

## Introduction

Asynchronous JavaScript and XML (AJAX) is a hot new technology on the internet [1]. AJAX allows the incremental update of portions of a web page without reloading the entire page. This has great performance benefits and provides a medium for developing rich internet applications. Accessibility concerns have arisen with the increased use of JavaScript and AJAX on the web. Advances in DHTML and its support by Assistive Technology (AT) vendors provide exciting new opportunities to create fully accessible AJAX applications. Currently there are several best practices and techniques that can be used to insure the accessibility of JavaScript and AJAX today.

## What is AJAX and How is it Used?

The ability to incrementally update a web page using eXtended markup language (XML) over hypertext transport protocol (HTTP) has been around for several years but did not gain wide appeal until recently when it began being used by internet companies such as Google [2], Amazon [3] and others. AJAX represents a combination of several web technologies and has become the “must-have” feature for web sites.

The key is the asynchronous interaction with the web server. Rather than the traditional web model of round trips to the server and reloading the entire page for each action, AJAX allows a “behind the scenes” interaction with the server to update portions of the page. This can provide a much richer and faster experience on the web. Using a combination of technologies such as HTML/XHTML, Cascading Style Sheets (CSS), JavaScript, Document Object Model (DOM) interactions, XMLHttpRequest object [4], and XML, a web author can create a highly interactive web application. Consider a traditional email application with a tree control representing a set of email folders on the left hand side of the page and documents contained in a folder listed on the right hand side. The user can select a folder and rather than waiting for the entire page to reload, the list of items in that folder is updated within the page. Another example is providing a preview of a mail’s contents in the lower half of a page without having to open the entire email in a new page.

## Accessibility Issues

First and foremost, AJAX relies on JavaScript so any AJAX application requires JavaScript being available in the browser. Most graphical browsers support JavaScript so this may only be an issue for some mobile devices or text only browsers. There is some concern that the World Wide Web Consortium (W3C) Web Content Accessibility Guidelines (WCAG) 1.0 [5] require that web sites function with JavaScript turned off.

Understanding the growth in rich internet applications, the WCAG 2.0 [6] under development has removed the restriction on JavaScript as long as sites meet all of the other WCAG 2.0 guidelines and are fully accessible. The Dynamic Accessible Web Content Roadmap [7] from the W3C Protocols and Formats group provides new standards in development to create fully accessible web components. Two emerging standards from this roadmap: States and Adaptable Properties Module [8] and the Role Taxonomy for Accessible Adaptable Applications [9] are designed to address the accessibility of these Rich Internet Applications. So, JavaScript is no longer the problem child of accessibility on the web.

Unfortunately, assistive technologies may not expect dynamic updates to web pages and can't always interact with an AJAX application. Users of screen readers may not get any indication that a portion of the web page has updated, even sighted users may not notice the updates. Unexpected changes in focus can also confuse users or force additional navigation. In addition, since pages are updated incrementally AJAX applications may not have distinct uniform resource identifiers (URI) that can be bookmarked. This can also lead to issues with the back button not functioning as expected to return to a particular page or state in the application. While there are accessibility issues to consider there are best practices that can help to make AJAX applications accessible

### **AJAX Best Practices**

First it is important to inform the user about the requirements of an AJAX application. Indicate that JavaScript and the XMLHttpRequest object must be supported. Use the HTML `<noscript>` tag to provide requirements information to users with no JavaScript capabilities or with JavaScript turned off. Provide a statement early in the page that indicates the use of AJAX technologies.

If the site relies on asynchronous updates to the page, give the user the option of receiving an alert when an update occurs [10]. This will help notify those users of AT who can't see the screen when an update has occurred. It also helps those who may use screen magnifiers and be scrolled away from the area which has changed. And, it assists those who may not easily notice the change due to cognitive or attention issues. This should be an optional notification that the user must elect to enable as the alert itself may prove more distracting for some users.

Do not automatically shift focus on the page when an update occurs. Changing focus without warning can be very distracting for some users, especially if there is no easy way to return to the previous position. For example, on a site that dynamically updates a series of stock prices located at the bottom of the page, a sudden shift in focus each time a price change occurs could make the page unusable. Do consider the use of color or a change in font size or weight to temporarily highlight the area which has updated. But, be cautious of causing a dramatic change or blinking which may distract some users.

Some changes in focus, however, may be appropriate. If the user has activated a button to check for new mail, it may be appropriate to move focus to the list of mail messages

when the update has completed. Make certain that the user is aware that an action is going to occur and expects to focus to change to the results.

Provide links to portions of the page which are dynamically updated. If the site relies on the update of several areas of the page at different frequency, provide a way to quickly navigate to each section. This set of links should be easily reached from the top or bottom of the page to make navigation to them quick and easily repeatable.

Replace data where possible rather than creating and adding new elements to the page. Not all AT can handle dynamic additions to a page via the DOM. When adding elements, append them at the end of a page or parent them to an existing element rather than the document itself. Be aware that adding navigable elements within the page may affect the tab order for AT users. Testing with AT is essential to ensure a fully accessible page.

The Dynamic Accessible Web Content Roadmap [5] explains how to incorporate role and state information into DHTML applications. This technology for DHTML accessibility sprung from a collaboration between IBM, the W3C and the Mozilla development community. As the W3C develops this new standard, IBM has implemented the stable pieces in Firefox and is offering expertise to help AT vendors fully support it. IBM is motivated to make this technology available as quickly as possible in order to improve the usability and accessibility of rich internet applications. Using this technology, web application developers can create fully keyboard accessible components that improve usability as well as accessibility. The additional role and state information can be translated by the browsers into a format usable by AT to provide more information about the function and state of a component. This technology maps completely to fully enabling AJAX accessibility.

### **Summary**

AJAX is an exciting new technology that is extending the web and allowing the creation of rich internet applications. Incremental updates to a page can improve performance and provide for more complex user interfaces and interactions via the Web. With careful design and development these applications can be more usable for all users, including those with disabilities.

### **References**

[1] Ajax: A New Approach to Web Applications. Jesse Garrett,

<http://www.adaptivepath.com/publications/essays/archives/000385.php>

[2] Google Suggest, <http://www.google.com/webhp?complete=1&hl=en> and Google

Maps, <http://maps.google.com/>

[3] Amazon A9 Search, <http://a9.com/-/home.jsp?nc=1>

[4] Very Dynamic Web Interfaces by Drew McLellan,

<http://www.xml.com/pub/a/2005/02/09/xml-http-request.html>

[5] Web Content Accessibility Guidelines 1.0, <http://www.w3.org/TR/WAI-WEBCONTENT/>

[6] W3C Web Content Accessibility Guidelines 2.0 , <http://www.w3.org/TR/WCAG20/>

[7] WAI Dynamic Accessible Web Content Roadmap 0.21,  
<http://www.w3.org/WAI/PF/roadmap/DHTMLRoadmap092305.html>

[8] States and Adaptable Properties Module <http://www.w3.org/WAI/PF/adaptable>

[9] Role Taxonomy for Accessible Adaptable Applications  
<http://www.w3.org/WAI/PF/GUI>

[10] AJAX and Accessibility, Peter Krantz, <http://www.standards-schmandards.com/index.php?2005/03/01/16-ajax-and-accessibility>