DB/C Newsletter

April 2006

News and Comment

AJAX is a new acronym for the not-so-new web browser applications technology that uses JavaScript and asynchronous HTTP requests in addition to standard HTTP and HTML. AJAX is a client-side only technology; server applications that create AJAX applications can be developed using any of the dozens of different languages and tools that are available for web server applications development.

AJAX is no magic bullet. There are applications for which AJAX is appropriate and there are applications for which it is not. AJAX is not simple or elegant. There are several disparate technologies that must be mastered to deploy an AJAX application. It's a long way from good old DB/C application development where everything is based on one language and development is done using one compiler and debugger! This month's newsletter provides an overview of AJAX pros and cons.

don.wills@dbcsoftware.com

AJAX

AJAX is a term that was invented in 2005. The acronym isn't a simple abbreviation for a few words, but is derived from some of the technology names it comprises: asynchronous, JavaScript and XML. The term AJAX was probably adopted was because it was cute and easy to say. AJAX actually encompasses these technologies -

HTML
HTTP
the asynchronous XMLHttpRequest enhancement to HTTP
JavaScript
DOM (the Document Object Model for web pages)
CSS (Cascading Style Sheets)

Traditional HTTP/HTML web pages are static. Applications using standard HTTP/HTML work by sending full pages of information between the browser and the server. A user fills out the fields on the web page and then clicks on the submit button which sends all of the information on the page to the server. The server then responds with a new page that replaces the old one. This is just like 3270 mainframe technology that has been in use for more than 35

years. Obviously, applications using only the synchronous HTTP/HTML request-response technology are a step backwards from today's interactive GUI and character-based applications.

The addition of JavaScript to the client side web page can add some level of intelligence to the user experience by allowing such things as simple field validation and intelligent field pre-fill. However, without access to the database, many simple actions such as file-lookup field validation are not possible. And unfortunately, JavaScript is a weakly typed, dynamic language meant for simple scripting, so JavaScript does not scale well. Managing and maintaining hundreds or thousands of JavaScript functions is problematic at best.

In 1999, Microsoft created the XMLHttpRequest command to provide an asynchronous method so that web page scripting code could communicate with the server without causing the web page to be redisplayed. The addition of this feature opened up many options for creating a better user experience including file-lookup field validation, context sensitive field pre-fill from the database, and many other useful user interface design patterns. The XMLHttpRequest function took a while to be adopted by non-Microsoft browsers and servers, but is now universally available.

The result is that programmers can now create web-based applications with functionality approaching that of rich desktop applications.

AJAX applications have some advantages when compared with traditionally deployed applications. First and foremost is the ease of installation and of upgrading AJAX applications. All of today's desktop computers come pre-configured with the software necessary to run AJAX applications - a web browser. Deployment of new releases of applications software is immediate with no extra work - new releases of software are automatically available whenever a user loads a new page.

However, AJAX applications have several drawbacks. Performance is typically inferior when compared with the same application deployed as a desktop application. Different browsers behave differently, so AJAX applications may not behave exactly the same way in all environments. There are differences in how JavaScript works on the different browsers, so JavaScript code is typically riddled with 'if' statements checking the identity of browser so that different code can be executed for each browser.

Another drawback is that HTML does not provide several of the basic controls that are found in GUI applications. An example is the menu bar with drop down menus. To add such functionality to AJAX applications, programmers use JavaScript code in the form of pre-developed or third-party libraries. Unfortunately, this means that various AJAX applications may or may not have similar user interfaces. One of the important advantages of GUI applications when compared with either AJAX or character-mode applications is that all GUI applications generally look and act the same when compared with other GUI applications. Because of the lack of a rich set of controls in HTML, complex AJAX applications may require more training and user assistance when compared with similar GUI applications.

From a developer's perspective, the biggest hurdle is the learning curve of the AJAX technologies. At least three languages must be mastered - HTML, JavaScript, and whatever server-side programming language you choose (Java and PHP are two of the more popular ones). Java and JavaScript have little in common - the syntax is similar, but that's about it, so if you know one you're only slightly ahead in learning the other. In addition, developers must

master the web page document object model (DOM) to be able to manipulate the web page from JavaScript. Unfortunately, DOM names and attributes are not fully standardized across different browsers. This makes building AJAX applications even harder, and that's also why some AJAX applications will only run on Microsoft Internet Explorer.

The decision to build AJAX applications is not a simple one. Many factors must be considered, from end user reactions to programmer tools to maintenance issues. Given the complexity and the moving target nature of the technologies, a prudent course of action may be to wait a year or two before making a decision about building an AJAX application.

DB/C DX Class Schedule

Class: DB/C DX Fundamentals
Date: June, 2006
Location: Woodridge, Illinois

For information, send email to admin@dbcsoftware.com.