Language Learning & Technology
http://llt.msu.edu/vol14num1/emerging.pdf
February 2010, Volume 14, Number 1
pp. 9–15

EMERGING TECHNOLOGIES: NEW DEVELOPMENTS IN WEB BROWSING AND AUTHORING

Robert Godwin-Jones
Virginia Commonwealth University

In this new decade of the 21st century, the Web is undergoing a significant transformation. Supplementing its traditional role of retrieving and displaying data, it is now becoming a vehicle for delivering Web-based applications, server-stored programs that feature sophisticated user interfaces and a full range of interactivity. Of course, it has long been possible to create interactive Web pages, but the interactivity has been more limited in scope and slower in execution than what is possible with locally-installed programs. The limitations in terms of page layout, interactive capabilities (like drag and drop), animations, media integration, and local data storage, may have had developers of Web-based language learning courseware yearning for the days of HyperCard and Toolbook. But now, with major new functionality being added to Web browsers, these limitations are, one by one, going away. Desktop applications are increasingly being made available in Web versions, even such substantial programs as Adobe PhotoShop and Microsoft Office. Included in this development are also commercial language learning applications, like Tell me More and Rosetta Stone. This movement has been accelerated by the growing popularity of smart phones, which feature full functionality Web browsers, able in many cases to run the same rich internet applications (RIA) as desktop Web browsers. A new Web-based operating system (OS) is even emerging, created by Google specifically to run Web applications. In this column we will discuss recent developments in Web browsing and authoring and what the implications may be for language learning. We will touch on what has changed since LLT columns surveying the state of the Web five years ago, "Ajax and Firefox: New Web Applications and Browsers" (2005), and ten years ago, "Web Browser Trends and Technologies" (2000).

WEB BROWSERS AS OPERATING SYSTEMS?

In Internet time five years ago is ancient history, so it may not be surprising that the discussion of browsers in 2005 included the recent release of Netscape 8, while 2000 marked the appearance of Netscape 6. This first widely-used browser finally gave up the ghost with Netscape 9 in 2008. But the big news in 2005 was the surge in popularity of a newcomer, Firefox, which actually arose from the ashes of Netscape in the guise of the Mozilla Foundation. In 2005, Firefox was making inroads in gaining market share over Microsoft's dominant market leader Internet Explorer (IE). Still, IE at that time was capturing about 84% of Web users (all such statistics are approximations). Today the percentage of IE users is considerably lower, around 63%. Firefox (now at version 3.5) continues to gain in popularity, usually attributed to its fast processing, security features, user-friendly interface, and the large number of add-ons available. Opera (from Norway) and Safari (from Apple) have considerably smaller percentages of the market. Yet these two browsers have more importance than their numbers suggest. They tend to support emerging Web standards early and implement experimental features, which often subsequently make their way into other browsers. The fact that IE no longer has a virtual stranglehold on the market has led Microsoft to be more responsive in adding user-requested features and, in some instances, in complying with Web standards. The current competitive browser market, in fact, is leading to much more development than was the case in recent years. Without competition in the browser market, it is not likely we would be seeing the host of new features now being incorporated into "modern browsers," often defined as Firefox, Opera, Safari, and Google's Chrome.

The Chrome browser was initially released in late 2008 and in the short time since, has become the third most widely used browser (after IE—versions 6,7, and 8 combined—and Firefox). This is in part due to innovative features such as the merging of the address bar with the search window, and a new, minimalist interface. Chrome also enhances security through technologies for isolating potential harmful content and

maintaining and checking against a database of malicious sites. This process isolation also adds more stability to the browser; if a Web site causes one tab to crash, the other tabs still keep working. One of the immediately noticeable user experiences with Chrome is the increased speed of JavaScript code execution. This is particularly important in enabling Web applications to run more smoothly. Chrome also introduces a new way of working with Web applications, allowing users to save them as icons to the desktop. When opened from the desktop, the browser window displaying the application shows only the title bar, not the rest of the browser toolbar. In this way, Chrome blurs the line between Web and desktop applications. Some analysts have argued that this kind of hybridity is the reason Microsoft until recently has been so slow in improving IE and expanding its functionality, in order to cement the prominent position it occupies in the OS and office application markets by stifling competition from the Web. Certainly, the existence of reliable Web delivered applications such as Google Docs, along with free or inexpensive Web file storage, offers a new alternative to the traditional desktop model. Increasingly, such applications take another step away from the desktop by enabling users to save documents to the "cloud" (i.e., to an Internet server).

Google has taken the next logical step in this evolution, by announcing the Chrome OS, slated to be released in 2010. It is designed to be a lightweight but fast environment for running Web applications. It is most likely to be used in netbooks (low cost small laptops without an optical drive) or other portable computing devices. The assumption—and business calculation—Google is making through this development is that enough of the functionality users want and need for their computing devices will be available through the Web, eliminating the need for a desktop OS altogether. Google is not the only company moving in this direction. Palm introduced in 2009 its new OS, called WebOS. Applications for new Palm devices are built using standard HTML, CSS, and JavaScript with some proprietary additions specific to Palm devices. The user interface is designed for a touch screen and features applications running as "cards" (shades of HyperCard) in a multitasking environment. Users click on a card to start up a program, flick left or right to move to other running programs and flick up and away to quit. The WebOS features easy social networking through its "synergy" function, which integrates all contact information from different sources and offers built-in access to Facebook as well as to a combined SMS/IM messaging window. The seamless integration of social networking services is a feature increasingly seen in browsers, both desktop and mobile. It reflects the enormous growth in the popularity of sites such as Facebook and services such as Twitter. Both the increased usage of such services and their gradual ubiquity may lead language professionals to think about leveraging student use of such services into language learning opportunities. There are already projects like the Facebook Language Exchange. With the rapid growth of the smartphone industry, this trend is likely to accelerate.

WebOS is built around the Web layout engine WebKit, originally developed by Apple from KHTML, for use in its Safari browser. Apple made WebKit open source in 2005 and since then it has been used in a number of browsers and mobile devices, including the Google Android platform, Palm devices, Nokia phones (running the Symbian OS), and the iPhone. The fact that all major smartphone manufacturers, except Blackberry (which reportedly will be upgrading its browser to WebKit), now use WebKit makes it easier for developers to create Web applications that will run predictably in different mobile environments, although some implementation quirks remain. This extends to desktop systems using WebKit as well as other devices such as tablets running Android or the iPhone OS. Of course it is also possible to create "native" applications which will run only on a specific platform. There are SDKs (software development kits) for both the iPhone and Android which, however, use different programming languages, Objective-C and Java respectively. While creating a native application may offer advantages such as tighter integration with device-specific features and a familiar, common user interface, this approach sacrifices the benefits of having the program run unchanged on multiple platforms. In fact, Web applications can be designed to look much like native applications. The iPhone allows, like Chrome, Web applications to be saved on the home screen as icons. Among iPhone developers there has been some

lively discussion on the choice of developing using Web standards or the proprietary iPhone SDK. The runaway success of the iPhone App store and the profits it offers developers might well point in that direction. However, given the diversity of devices now available, it would seem to be best practice for developers of language learning programs to use open Web standards. The exception might be situations in which a particular device or OS has been selected for exclusive use, as in campuses making the iPhone or iPod Touch available to all their students.

HTML 5 ON THE RISE

What makes it possible for Chrome or the iPhone to make Web applications available locally, as if they were native applications, are new features that are part of the upcoming HTML 5 standard. In fact, a major element in the design of HTML 5 is to improve the delivery of Web applications. This is apparent from the name and make-up of the group which initially pushed for the standard, WhatWG, the Web Hypertext Application Technology Working Group, founded by representatives from Apple, Mozilla, and Opera. From the perspective of 2005, the fact that a new version of HTML was being developed at all would be a surprise, as the World Wide Web Consortium (W3C) had planned on HTML being phased out, to be replaced by the XML variant of HTML, XHTML. XHTML 1.0 became a W3C recommendation in 2000. HTML 5 was first developed independently of the W3C by the WhatWG group but since has been incorporated into a new HTML working group at W3C. The call for the continued use and further development of HTML was based on the desire to keep the fundamental language of the Web as simple, backwards-compatible, and forgiving as it had been since the early days. The stricter rules of XML make this less easily doable in that language. In order to satisfy both groups, and to insure compatibility with existing XHMTL pages, the current plan calls for development of both HTML 5 and a corresponding XHTML 5. The first full candidate release of HTML 5 is scheduled for 2012 but many of its features have already been adopted in browsers, with Firefox, Opera, Safari, and Chrome offering the most support. HTML 5 introduces a number of changes in HTML tags, eliminating some (such as align, bgcolor, frameset) while adding many additional tags and features. The deprecated tags will continue to be supported by browsers (what a mess the lack of support for frames would be), but to be valid HTML 5, dropped tags will not be able to be included. One of the major directions for the changes in tags is to allow for more semantically meaningful mark-up of pages. New elements, which supplement the ubiquitous “div” tags include header, footer, article, and aside, all of which, along with other new tags, should make it easier to identify (and search) the structure and contents of Web pages. This is also likely to make page rendering faster.

As could be expected, a number of the new features in HTML 5 are designed to offer better support to Web applications. One of the limitations that Web applications have had in comparison to their desktop cousins, has been the limited ability to store data short- or long-term on the local machine. HTTP cookies offer only limited data storage (20 cookies of 4 KB each per domain). HTML 5 introduces DOM storage (also called Web storage) which, like cookies, allows saving of data into associative arrays (key/value pairs) but with much larger capacity (up to 10 MB per domain). HTML 5 also offers Web application the ability to cache needed files locally (listed in a manifest file) which, assuming needed run data has been saved, can then be run even if an Internet connection is not available. This is, for example, how Gmail works off-line. The process is used by the iPhone OS and Chrome to be able to create shortcuts to saved Web apps.

In the 2005 column, I discussed the arrival of AJAX (asynchronous JavaScript and XML) as a new means to create Web applications. The JavaScript functions work in the background to fetch new data from a server, which allows content on a Web page to be updated without the need for the page to re-load. The XMLHttpRequest object, originally a proprietary feature added to IE, allows this to work. Given its wide-spread use, this object (along with other popular Microsoft introductions such as innerHTML and contentEditable) are being integrated into the HTML 5 standard. To further improve Web applications,

"Web workers" are also part of the proposed standard. Web workers are not, as the name may suggest, techies building Web sites, but a means to allow JavaScript code to run in parallel rather than sequentially. These separate background threads allow AJAX pages to run more efficiently.

In comparison to 2005, performance of AJAX pages is already enhanced in browsers today through faster execution of JavaScript. From being a much maligned add-on to Netscape 2 in 1995, JavaScript has become the essential glue that allows page elements to interact with one another and with server resources. The importance of JavaScript for Web designers has led to the creation of a number of JavaScript code libraries (especially for use in AJAX), such as jquery. The OpenAJAX Alliance has been created to try to achieve some level of interoperability among the libraries. Google has also been working in this direction, through creation of an AJAX Libraries API, which makes it very easy to use the most popular JavaScript libraries for Web applications. With just a few lines of code, developers can easily add very powerful JavaScript functionality from a great variety of libraries. The libraries themselves are hosted on Google servers (which pledges to keep them "indefinitely") and are regularly updated. The increased efficiency of JavaScript execution in all browsers, but especially in Firefox and Safari, are orders of magnitude better than in 2005. This, too, allows operation of Web applications to more closely resemble desktop applications. Another new HTML 5 feature, cross-window messaging also points in this direction, allowing text messages to be sent (normally in the background) from the current window to other windows.

WHAT DOES HTML 5 OFFER LANGUAGE LEARNING?

Many educational resources for the Web are created in Adobe Flash, a proprietary format that uses a browser plug-in to play movies and interactive content. While Flash is widely supported it does not run on all platforms, and in particular not on some mobile OS's. This is true as well for Microsoft's competitor to Flash, Silverlight. Each program can be used to embed interactive content which is able to communicate with other elements of a page through JavaScript. Both Silverlight and Flash, through its rich Internet application client Flex, also allow Web applications to run on the desktop like native applications. However, integrating Flash or Silverlight objects into a Web page is not as seamless as it is using standard HTML elements. There is also frequently an issue with slow performance and even crashes when running Flash, particularly on Linux and Mac OS. Nevertheless, both technologies will certainly continue to be important for Web developers, including language-learning professionals.

HTML 5 introduces alternatives to Flash or Silverlight. New "audio" and "video" tags allow integration of these media in a similar way to how the image tag currently works in HTML. As a regular part of the default document object model (DOM) of the page, an HTML media object offers easier options for interacting with other elements on the page, as can be seen in on-line demos. [Note: HTML 5 examples cited here will need to be viewed in a recent version of Firefox, Safari, Opera, or Chrome]. The audio/video tags also simplify the task of creating custom playback controls. This feature is clearly of considerable potential interest for developers of language learning programs which incorporate video and audio, enabling, for example subtitles to be added easily (through a text file referenced by the video tag) turned off or on by the user. The audio/video playback capabilities of HTML 5 will also allow for synchronized multimedia. Of even greater interest as an alternative to Flash development is HTML 5 support for Canvas, originally introduced as a proprietary object in Apple Safari browser. Canvas is used for creating graphics through use of JavaScript and CSS code, allowing graphical elements to be created and changed on the fly. The ability of Canvas to paint directly to the page has made it a compelling alternative for game developers, as seen in on-line prototypes which demonstrate some of its capabilities. Simulations can also be created in Canvas, as a physics experiment illustrates. Sound can also be added. One could envision using Canvas to create a simulated environment for culture or language applications, such as building scenarios for greetings, shopping, buying a ticket, etc. While Canvas is currently supported in most browsers, for IE a work-around is necessary. Many implementations of Canvas use a

powerful JavaScript library, processing.js, which makes the somewhat complex operations needed to create images or animations much easier. It is a powerful tool for making visual representations of data, building custom user interfaces, or developing Web-based games. Canvas is supported in WebKit , so applications incorporating it should run on mobile WebKit-based browsers. The issue of compatibility also effects currently the use of the video tag, with different browsers supporting different video encoding formats.

It might be mentioned in this context how tightly Internet giant Google has tied its wagon to HTML 5, likely forcing other companies eventually to follow suit in terms of HTML 5 support. The most compelling evidence of Google's commitment is Google Wave, a rich collaborative environment, created using HTML 5 technologies. Google Wave breaks down the barriers between different kinds of services and document types. It features a window displaying a set of documents (a "Wave") which can include virtually any kind of content. All documents are stored on a central server. If a group member of a Wave creates a new addition to the Wave or edits an existing resource, this will immediately appear in real time in the Wave display of all members of that group, including, amazingly, what is being typed letter by letter. In this way, the system, works like a combination of email and instant messaging. Because the history of the Wave is maintained in all its detail, it also works as a rich collaborative environment, like a blog or wiki. Documents can also be uploaded and shared, with a change log to see revisions and the ability to revert to previous versions. In that way, it provides much of the functionality of versioning in office applications. Additional functionality is possible through adding extensions to Wave; currently there are half a dozen language tools available as extensions. There is currently a Calico Wave exploring what the application might mean for language learning, including the question of whether the use of Google Wave might even serve as a replacement for a learning management system such as Blackboard. Some of the uses explored include the use of a voice recording/playback extension, a translation robot, and video conferencing. Google Wave is still in beta so it is too early to tell whether it will be as transformative as some claim. At a minimum, its advanced features demonstrate in a dramatic way the high degree of interactivity and collaboration becoming possible on the Web.

Other HTML 5 elements provide additional intriguing possibilities for language learning. New HTML form fields offer over a dozen new modes of interaction. Drag and drop is now possible not only within the browser but extending to the computer desktop. This includes the ability to drag and drop multiple files at a time from the local computer to be sent to a server. These additions take a further step in allowing Web applications to function in similar ways to desktop programs. They make it more likely that language-learning programs, such as intelligent language tutors, could be coded in HTML 5, adding the benefits of having them available on mobile devices. Another new HTML feature, geolocation adds the ability to track user locations. This makes it easier to create services like Google Latitude, which finds and indicates on a map the locations of members of a specified group. All of this new functionality could be of significant interest in creating interactive Web pages for grammar practice or collaborative environments for group interactions. While currently not supported in all browsers, forward-looking developers would be well–advised to follow HTML 5's gradual adoption, in order to consider taking advantage of new features as they become widely supported.

RESOURCE LIST

Demos of HTML 5 features

- A form of madness–Dive Into HTML5
- Animations, transitions and 3D transforms - Morphing power cubes
- Bespin - Code in the cloud - From Mozilla Lab

- Box2DJS - Physics engine for JavaScript
- Canvascape – "3D Walker" - Ben Joffe
- CanvasPaint
- Choose your own HTML 5 adventure
- Chrome experiments - Home
- ContentEditable - HTML5 demo
- HTML 5 demos and examples
- HTML5 Canvas and audio experiment
- HTML5 Canvas experiment
- Javascript Wolfenstein 3D - From Nihilogic blog
- Jigsaw puzzle by Raymond Hill: An HTML5 <canvas>-based jigsaw puzzle
- O3D Beach demo - YouTube HTML5 demo
- PaintWeb integration examples
- physicSketch
- Poster Circle
- Web forms 2 demo page by Shwetank Dixit
- WebKit HTML 5 SQL dtorage notes demo

HTML 5 and Web Standards

- A preview of HTML 5 - From A list apart: For people who make websites
- An introduction to the Canvas element - From Bright Hub
- Bespin and the open Web - From Google I/O developer conference
- Canvas primer - Example: Using gradients
- DOM storage - From Wikipedia, the free encyclopedia
- Google bets big on HTML 5: News from Google I/O - From O'Reilly Radar
- Hoa: Offline applications with HTML 5 - From msc mobile blog
- How HTML5 will change the way you wse the Web - From Lifehacker
- HTML 5 canvas - the basics - From Opera Developer Community
- HTML 5 differences from HTML 4
- HTML 5 reference
- HTML 5: Could it kill Flash and Silverlight? - From InfoWorld
- HTML5 (including next generation additions still in development) Google goes HTML5: Demos experimental version of Gmail - From ReadWriteWeb
- HTML5 and the future of the Web - From Smashing Magazine

- HTML5 features at a glance - A selection of support features
- HTML5 file API brings drag-and-drop uploads to the Web - From Webmonkey blog
- HTML5 JS APIs
- Javascript geolocation using Google AJAX APIs - The why and the how
- Microsoft joins HTML 5 standard fray in earnest - From Business Tech - CNET News
- New elements in HTML 5: Structure and semantics - From IBM developer works library
- On HTML 5 drag and drop - From Alert Debugging
- Using files from web applications - From Mozilla Developer Center
- Watch YouTube videos without Flash in HTML5 - From The NeoSmart Files
- Yes, you can use HTML 5 Today! - HTML & XHTML Tutorials

JavaScript

- Google AJAX Libraries API - Google code
- JavaScript engine speeds - From John Resig blog
- JavaScript gaming - The best JavaScript games on the web
- Processing.js
- The future of JavaScript engines: Replace them with JavaScript compilers - From Shore Street blog
- Top JavaScript and AJAX Libraries

Language Learning

- 50 iPhone apps to help you learn a new language - From Online College Degree
- Cheat sheet: Hints and tips for Google Wave - From The Shiny Wave
- Google Wave extensions list
- Google Wave: 5 ways it could change the Web - From Mashable: The social media guide
- Language Exchange - From Facebook
- Language learning applications for smartphones, or small can be beautiful
- Smartphones drive language learning innovation - From Physorg.com
- The #iPod touch and the #iPhone will change language learning - From The Linguist List blogs

Web Browsers

- Browser statistics - From W3schools.com
- Google's Chrome grabs No. 3 browser spot from Safari - From Computerworld
- Testing mobile browser compatibility–the beginning - From QuirksBlog
- Usage share of web browsers - From Wikipedia, the free encyclopedia