Web Development Skills for Information Architects

By Judith Wusteman
Published in UXMatters, June 23, 2014

Abstract
Some information architects can merely propose information architectures, but the most useful information architects have skills that enable them to be actively involved in Web development.

What Web skills does a good information architect need? In this article, I’ll propose a set of Web skills that a graduate with a Masters in Information Architecture would ideally possess.

Well-rounded individuals

“Get well-rounded individuals. Go for quick-learning generalists over ingrained specialists…. We’ll never hire someone who’s an information architect. It’s just too overly specific. With a small team like ours, it doesn’t make sense to hire people with such a narrowly defined skillset.”—from Getting Real [1]

This quote is from Getting Real, the popular book on how to build a successful Web application, which goes on to say, “Small teams need people who can wear different hats” and “Everyone should have an idea about how to architect information (whatever that may mean).” [1] These are controversial statements, but I suspect that many information architects (IAs) would agree that there’s some truth in them.

Within the information architecture community, there’s an awareness that IAs need to avoid becoming precious about what they’re prepared to do; that they need to be ready and able to get involved in activities beyond the basic information
architecture work of organization, labeling, and navigation. The question is: how much more should they be able to do and in what areas?

Donna Spencer [2] suggests that, in addition to defining information architectures, many information architects also “do strategy, content writing, interaction design, and production.” So it’s not just in the area of Web skills that IAs need to be flexible.

But what about Web skills? Which would be the most useful and why? In this article, I’ll suggest the Web development skills that I think would make IAs most employable. My proposals aren’t based on a formal study, but on my understanding of the field of information architecture and on informal communications with former students and potential employers.

Of course, metadata and Webometrics are of central and growing importance to all aspects of Web development in general and information architecture in particular. However, these topics are beyond the scope of this article.

Some Pragmatic Proposals

My proposals are pragmatic and take into account the fact that many graduates with a Masters in Information Architecture will end up in roles in which they'll design some information architectures, but that may not be their main focus. I’ve also attempted to take into account the broad range of backgrounds and technical abilities among typical students in information architecture programs.

For example, the ENS-Lyon Masters “is open to students from all disciplines … also available as part of continuing professional development to: persons performing managerial functions wishing to acquire expertise in both IT and digital documentation; persons performing technical functions and wishing to move into positions of engineer or project manager in information architecture.” [3] While this is very broad, teachers of information studies / science are familiar with trying to convey technical Web development skills to students with very varied technical abilities. Most of the time, it works; sometimes it doesn’t. What I’m proposing won’t suit all students, so flexibility is necessary. My proposals also assume that a student
is studying for a Masters degree in Information Architecture rather than just taking a single information architecture course in a larger program of study.

**HTML5**

With the advent of HTML5, there is an awful lot more to understanding HTML than just being familiar with the concept of markup. HTML5 is a lot bigger and a lot more complex than XHTML. Many sites are still using XHTML or even HTML4. The typical XHTML site involves HTML, some CSS, and a few JavaScript calls. We’re still in the early-adoption cycle for HTML5, but it’s already clear that the Web sites and applications that we’ve become familiar with over the last decade don’t represent the future of the Web. Prior to HTML5, Web sites have largely been information-dissemination conduits; most of the content on such sites is similar to that in an organization-based magazine or brochure.

But HTML5 has exploded beyond those bounds. Its focus is no longer static markup. HTML5 is highly dynamic, highly interactive, and often, highly social and collaborative. Although HTML5 is a lot bigger and more complex than XHTML, in some ways, HTML5 is easier than XHTML. No longer having to explain HTML4 or XHTML doctypes will be a relief to many teachers. Choosing to continue designing Web sites the way we always have would be business as usual, but would miss the point of HTML5 and related Web standards and ignore their potential. Old approaches to Web development are really not what we should be teaching the next generation of IAs. Instead, we should be promoting the use of related Web standards, including the following:

- Web Workers
- Drag and Drop
- File API
- Connectivity: Web Socket/Server-Sent Events/PeerConnection
- Geo Location
- postMessage
- Web Storage, IndexedDB, Web SQL Database
- Web Audio
Full-screen Web Components (HTML Templates, Custom Elements, Shadow DOM, HTML Imports)
CSS—FlexBox, Animations, Web Fonts, Multi-column, transformations and transitions, regions and exclusions, media queries, calc()
Microdata, form fields, form validation
Web Intents
Canvas, WebGL, advanced SVG & SMIL support
WebRTC
EventSource <menu>, <meter>, <progress>

**HTML5 = HTML + CSS + JavaScript + Related Standards**

The seismic waves that are the mobile Web and the social Web are bound to ensure that it’s not business as usual in the Web development world. HTML5 = HTML + CSS + JavaScript, but it’s also all of the related standards that I listed earlier. Initially, it was assumed that all of these extra features would be part of the HTML5 standard, but as the full extent of the new Web emerged, it became clear that there was far too much to encompass in one standard and the W3C began splitting things off from HTML5. So now there’s an entire family of standards. A good IA should understand this family of standards and fully realize the implications of what they make possible.

**IA as Change Agent**

The best IAs are change agents, helping others to move forward. If they’re to adopt this role, IAs must be completely *au fait* with what’s possible in Website design. HTML5 is—and should be—a huge challenge for everyone in Web development. The world of Web development is changing exponentially. Inevitably, some people are going to be left behind. The challenge is to ensure that the students emerging from academic information architecture programs aren’t among this group.
It’s not just the IAs who must up their game. HTML5 is a challenge for everyone in Web design and development. For example, visual designers have, in the past, concentrated on how things look. But with new, highly interactive Web pages, they’re going to have to extend their focus to how things behave. In other words, their role is morphing into that of interaction designers. Indeed, as the number of skills that Web development requires increases, fewer and fewer organizations will be able to afford a different member of staff to cover each role. So every employee has to wear even more hats than they do at present. (Having said this, I would not suggest that IAs take on the role of project managers—or, in agile parlance, Scrum Masters. In most cases, that wouldn’t be the best use of an IA’s skills.)

If an IA understands this sea-change in the roles of Web sites and applications, as well as Web development team members, they can support and advise other members of their team, leading the team toward the use of more dynamic interactions. IAs should sufficiently understand the potential of HTML5 that they can at least suggest to developers and visual designers what is possible for them to achieve. Alternatively, they should be able to work constructively with interaction designers, if the team includes someone with that title. You might think that all of this is someone else’s job, but it’s impossible to separate the information architecture from the way users interact with it. And, as I mentioned before, we need to be pragmatic and train IAs to be as flexible and useful—and therefore, as employable—as possible.

**Learning Web Development Skills**

So, in practice, what does this mean for the syllabus of a Masters in Information Architecture?

While, until recently, you might have gotten away with one course on Web publishing, in which students would learn the basics of XHTML and CSS markup and possibly a bit of basic JavaScript, I propose that a Masters in Information Architecture now needs three courses in Web development skills—each worth five ECTS (European Credit Transfer and Accumulation System) credits: [2]

- HTML5 and CSS3 markup
object-oriented JavaScript—plus a popular JavaScript framework such as jQuery or Dojo
- the standards in the HTML5 family

Course 1: HTML5 and CSS3 Markup

First, considering the course on markup, I propose that HTML and CSS be taught from scratch. Avoid using WYSIWYG Web-page editors. Instead, stick with text editors like Notepad or more powerful Web code editors. In my experience, this is the only way to ensure that students really understand the markup—and they’ll need to understand it in the workplace. If the JavaScript course requires the use of an integrated development environment (IDE) such as Netbeans, students can use the same IDE when editing HTML and CSS.

This module should also cover the basic tenets behind HTML: document hierarchy, how to identify that hierarchy through markup, how to document hierarchies using trees, the separation of presentation and content, and building content for reuse. Focusing on these tenets in this course ensures that the students build transferable skills that will outlive the current versions of popular Web code editors.

Of course, familiarity with a Web code editor is useful—particularly if the use of that editor is ubiquitous—such as for Dreamweaver, which has a sufficiently steep learning curve that it’s helpful to have spent time getting to know it before using it in a work setting. But, in my experience, if you want to ensure a deep understanding of HTML and CSS, there just isn’t time to include how to use a Web code editor in the same five-credit course. Besides, students often can’t resist using such an editor to avoid having to come to grips with the complexities of markup. So, if you want to cover editors, do it in a separate course.

For those of us who have been teaching HTML for years, it’s tempting to save time by simply tacking a lecture on HTML5 onto an XHTML course. But doing this would prevent your teaching HTML5 as a coherent whole. Of course, it’s also important for students to appreciate where HTML5 has come from, and they will
likely need to understand and edit legacy documents in earlier versions of HTML and XHTML.

Course 2: JavaScript

Many Web sites of the future will use JavaScript heavily. So budding IAs need experience in hands-on development of object-oriented JavaScript, plus at least one of the key JavaScript frameworks. A good choice for the latter is jQuery, currently the leading JavaScript framework. An alternative is Dojo. Familiarity with the composition of effective user interfaces using widgets—for example, jQuery widgets—is very useful. Even more important is an awareness of when you’d need to develop a custom widget to meet user needs. An acquaintance with the issues surrounding graceful degradation, shims, polyfills, and liquid design is important. A quick introduction to CoffeeScript might also be useful.

Course 3: Related Web Standards

The third of our three Web-development skills courses should provide the expertise that would allow IAs to discover the potential of the continually evolving family of Web standards. Earlier, I listed some of the main non-markup specifications in the HTML5 family. They’re all essentially client-based standards. So this course could focus on client-based standards rather than server-side development. Next, I’ll describe just a few examples that show the potential of these standards.

The Canvas

The canvas, the HTML5 drawing surface, is one of the central features of HTML5. It can be in 2D or 3D, you can use JavaScript to draw on it, and the visual results can be stunning.

Web Audio

The Web Audio standard facilitates the creation and modification of real-time audio. Web Audio provides a good example that illustrates the nature of the new HTML5
world: There isn’t a single HTML tag or CSS property associated with it. It’s composed entirely of browser objects and JavaScript.

**WebGL**

It is important for IAs to understand the WebGL standard because it is central to the development of the next generation of realistic 3D games and visualizations. The move from Flash to HTML5 is happening at a fast rate. For a short period, we’ll still need to develop shims in Flash for browsers that do not support HTML5. But, in general, new information architects can simply skip the Flash generation.

**Local Storage**

HTML5 Websites can store large amounts of data in the client. For example, Google Calendar lets users access calendar events even when they’re disconnected from the Web. This new local-storage capability is a big change from the model to which Web developers have been accustomed until now, where any user data generally gets uploaded to the server.

Web applications that remain useful even when they’re disconnected from the Web are becoming increasingly important with the growth of tablets. Indeed, it’s the explosion of HTML5-ready mobile devices and tablets that has been responsible for the rapid move to HTML5. The mobile Web is also having great impact on the evolution of information architecture. As companies reimplement their Web sites for optimal presentation on mobile devices and tablets, and new sites target these devices, the work of the next generation of IAs will increasingly focus on such devices.

**Server-side Development Environments**

In addition to learning the new generation of client-side HTML5 standards, IAs also need to be aware of modern server-side development practices—such as the use of client-side widgets that are bound to server-side RESTful services—instead of traditional JSP-style dynamic Web pages. It would also be advantageous to have a general awareness of common server-side development environments such as Java, .NET, LAMP (Linux Apache MySQL PHP/Pearl), and node.js. Where should this
topic fit into my three proposed courses? Perhaps just one lecture at the end of the JavaScript course.

**Agile Development Processes**
An understanding of agile development processes should permeate all of the courses for a Masters in Information Architecture.

While more traditional development methods ignore the possibility of changes in a project’s requirements, agile development methods attempt to respond to changes. They involve a rapid succession of incremental software releases, or *iterations*; early coding, lean documentation, close collaboration between developers and users, and rapid feedback. Agile teams avoid defining fixed roles for specific team members as far as possible.

The five-credit Information Architecture course that the UCD Dublin iSchool offers is based on a project in which a student imagines that he or she is an information architect on a team that’s creating a Web site using agile methods. This allows students to gain an appreciation for and familiarity with agile methods.

While in an agile development environment, developers may focus on development issues 100% of the time and visual designers may focus on design 100% of time, to really understand what’s possible and keep their skills fresh and up to date, IAs should do some Web development. They may take on just some of the simpler development tasks, for perhaps only 10% of their time, but their being actively involved in development is the ideal.

**Life-long Learning**
People who want to be IAs need to understand that earning a Masters in Information Architecture would just start them on the road to becoming excellent IA practitioners. They need to be prepared for life-long learning in all aspects of their craft, but particularly in terms of their Web development skills. The Web is rapidly and constantly evolving, and IAs need to keep up with these changes. Part of life-long learning in relation to Web development skills involves keeping abreast of the latest best practices.
IAs need to know what’s possible and have at least a general idea of how to go about making it happen. In other words, they need to understand everything that a developer understands. But they also need to be aware of many other issues—notably everything to do with usability—so must know more than anyone else on their team. This is a huge demand and, inevitably, at the start of their career, IAs won’t know more than everyone else on a Web development team. But they should recognize that their career path should, ideally, lead them in that direction.

Note—This article is an updated version of an invited lecture that I presented at *Architecture de l’Information - Colloque International*, Écoles Normales Supérieures de Lyon, November 19–20, 2012.

**Endnotes**


