

The Web Insider . . .

Web Wizards: Engineers, Artists, and Librarians*

Anna Belle Leiserson**

Ms. Leiserson introduces a new Law Library Journal column about the role librarians can play “inside” the World Wide Web—helping to shape and develop it. For her initial article, she describes what is involved in developing an outstanding Web site, emphasizing in particular the important part librarians can play in creating the information architecture of the site.

Introduction—A New Column

¶1 Welcome to the first in a series of articles for *Law Library Journal* in which I will explore the Web from the inside. Typically we librarians articulate an outside, consumer, or end user perspective when reflecting on and working with the Web. We review particular sites, analyze the best search engines, or tailor our Web bookmarks. But the library community has another major role to play that is inside the Web—actually shaping it and actively participating in what is now most frequently called Web development—and that role will be the focus for “The Web Insider.”

¶2 As Web developers, we can operate in two modes. First there is what I would call *micro* Web development. This encompasses the vast majority of Web development—that is, the creation and maintenance of specific Web sites. Second is *macro* Web development, helping to mold the structures and standards behind the Web itself. For librarians, the most obvious example of the latter is the role we play in metadata and our work on the Dublin Core. But can our role go beyond this niche? What is our part in indexing the Internet? What is it about the current hot topic of the moment—“usability”—that seems so familiar to us? Where do librarians fit in the open source movement? Why does the public view the Web as a library, and what does this mean for us? What is the Web, really? These are the type of issues I will be considering in future articles. For this initial column, however, the primary focus will simply be on what is involved in developing a Web site. Think of it as “Webmastery 101.”

* © Anna Belle Leiserson, 2002.

** Webmaster, Vanderbilt University Law School, Nashville, Tennessee.

Creating a Web Site

¶3 It is actually quite fascinating to listen to users reflect on how a Web site is created. One gets the sense that they believe webmasters, with our dungeon-and-dragon job title, wave some sort of programming wand, use arcane knowledge found in secret Web sites, and bibbety, bobbity, boo—a Web site appears. If you push for specifics, chances are users will say that what is required is knowing some “funky code.”¹ Funky code is the modern-day equivalent of incantations and magic words.

¶4 While there is, of course, some truth to the belief that code (HTML specifically) is what it’s all about, that only captures about one-fifth of the picture. Why one-fifth? Because, in essence, five professions actually converge during the creation of most good Web sites.

¶5 HTML is the standard that made the Web possible and is thus the heart of the Web. Without it there would be no Web. Close kin to HTML are Perl, PHP, Javascript, ASP, and a host of other codes with peculiar acronyms. All of these various codes have their roots in programming and engineering. So that is the first profession involved. However, as most programmers will tell you, knowing these codes is not enough to build a good site. After codes, the element required in a Web site that is most apparent to users is that of aesthetics and design. The Web can be so utterly visual. It was only when it moved beyond text-based browsers, such as Lynx, to a graphical interface, such as Mosaic (later to become Netscape), that the Web exploded into our collective consciousness. So this is the second profession required: the graphic artist and designer. A more sophisticated user might also acknowledge that content is important, but, in truth, content is king. Without good content, a Web site is nothing. So you can easily identify the third profession involved—publishing, including writing.

¶6 The remaining two jobs required for creating a good Web site are much less apparent: administration and information architecture. For our purposes, the last is particularly important, because that is where librarians can and should fit in. The fact that this function is not obvious relates to where we are right now in the development of the Web, including our own self-perceptions. Ironically, bad architecture is frequently what bothers users the most. The catch is they do not realize what is bothering them. Thus, we librarians need to assert our knowledge and join the engineers and artists in molding the Web. And even if we do not aspire to such lofty goals, at a bare minimum we need to improve our library Web sites. (The average library Web site needs significant revamping, but that is another column.)

¶7 It should be clear even from this brief introduction that there is a tremendous amount of knowledge and work involved in good Web site creation—that a site of any size is beyond the ability of any one person. Coders have their place; so

1. Quoting a close relative of the author who prefers to remain anonymous.

do information architects. But all of us can easily grasp the fundamentals of Web development. All that is needed is to break the process into manageable pieces. So the rest of this article will be devoted to just that: the nuts and bolts of developing a Web site, including creating and maintaining it.

Webmastery 101

¶8 The “Checklist for Web Site Development” contained in Appendix A outlines the basic steps involved in developing a Web site. The checklist is broken into the five primary job types described above, although there is a certain amount of cross-pollination among the jobs. The webmaster’s responsibility is to coordinate these jobs, so the checklist is done from that perspective.

Job One: Administration

¶9 The process of creating or redesigning a Web site typically begins in an administrative mode—defining the goals and getting the key players in place. A fundamental part of this job that might seem obvious but which, in fact, can be amazingly slippery is identifying the core group that has authority over the site. For example, it might seem clear for a law firm Web site that the senior partner would have a role, but does this person know and understand that? If not, who plays that role, and will this shift in responsibility work in the long run? These can be difficult questions—harder, really, than HTML code—and may not be answerable immediately (or ever), but the more clarity you can have at the outset, the better.

¶10 This core group will be the most likely source of basic information—what content needs to be covered in particular. They can also tell you if there are guidelines to adhere to. If there are not any guidelines, it is probably a good idea to draft some basic ones at this stage (covering things such as how to note copyright on the Web site).

¶11 This is also the stage to choose the tools you will be using, in particular Web-authoring software. Good choices are Dreamweaver² and GoLive.³ FrontPage can also work, but there are more complications associated with it. (Though generally considered easier to learn, it produces nonstandard code.⁴) Other key tools are graphics software (perhaps Photoshop⁵ or Paintshop Pro⁶), a

2. For more information, see MACROMEDIA DREAMWEAVER 4, at <http://www.macromedia.com/software/dreamweaver/> (last visited Nov. 5, 2001).

3. For more information, see ADOBE GOLIVE 5, at <http://www.adobe.com/products/golive/> (last visited Nov. 5, 2001).

4. For more information, see MICROSOFT FRONTPAGE, at <http://www.microsoft.com/frontpage/> (last visited Nov. 5, 2001).

5. For more information, see ADOBE PHOTOSHOP 6, at <http://www.adobe.com/products/photoshop/> (last visited Nov. 5, 2001).

6. For more information, see JASC PAINT SHOP PRO 7, at <http://www.jasc.com/products/psp/> (last visited Nov. 5, 2001) (“create and optimize Web graphics with built-in Web tools and artistic drawing and text tools”).

text editor (NoteTab Light⁷ is freeware and a good choice), and file transfer software (such as CuteFTP⁸ or WS_FTP⁹). A link-checker such as Xenu¹⁰ (also freeware) and a statistical analysis tool such as WebTrends¹¹ might also be helpful. In addition you will have to have the same browsers as your users, so you can replicate the way they see the site. Typically this means having Netscape 4 and 6, and Internet Explorer 4 and 5.

¶12 Another basic is to define the audience for your Web site. Will there be users from all over the world or just staff on an intranet? Is your primary audience current students, librarians, attorneys, or other groups? Such distinctions make an enormous difference in the site design. Communication is the key here—working, of course, with the core group, but also a Web committee (if you have one), key staff, and possibly the previous webmaster. A survey might help, though this is something to consider carefully before undertaking.

¶13 If there is already a Web site running, an excellent statistical resource is its “log files.” The Web server automatically accumulates data about every section of the site accessed by any user. These log files typically record the user’s IP number, the date and time, the page accessed, the referring page, and the browser and platform used. No one in their right mind would want to look at the raw data,¹² but fortunately, software such as WebTrends will take this data and crunch it into meaningful numbers. In particular, it shows what parts of the site are used the most and the least, the browsers used to access the site, and how people are finding the pages—all exceedingly useful information for understanding your audience.

¶14 With these basic pieces mapped out, you can actually start to draft the site, draw up a timetable, and decide what staff will be needed. The ideal team for creating a large Web site would have people for all of the following roles: the coordinator (or webmaster), designers (or art editors), photographers, A/V technicians, writers, HTML coders, programmers, and copy editors. The reality, of course, is that the job usually falls to one or two people—who then take on the aforementioned sorcerer-like mystique.

7. For more information, see NOTE TAB, at <http://www.notetab.com/> (last visited Nov. 5, 2001).

8. For more information, see GLOBALSCAPE, CUTEFTP, at <http://www.cuteftp.com/> (last visited Nov. 5, 2001).

9. For more information, see IPSWITCH WS_FTP PRO 7.0, at <http://www.ipswitch.com/> (last visited Nov. 5, 2001) (“a file transfer application that is used to transfer files between your local PC and a remote FTP server”).

10. For more information, see XENU’S LINK SLEUTH, at <http://home.snafu.de/tilman/xenulink.html> (last visited Nov. 5, 2001) (“a spidering software that checks Web sites for broken links”).

11. For more information, see WEBTRENDS, at <http://www.Webtrends.com/> (last visited Nov. 5, 2001).

12. Here is a sample: *129.59.6.41 - - [23/Sep/2001:05:46:45 -0500] "GET /library/index.html HTTP/1.0" 200 13034 "/admiss/tests.html" "Mozilla/4.0 (compatible; MSIE 5.0; CS 2000; Windows 98; DigExt)"*. What this means is that someone whose computer was assigned an IP number of 129.59.6.41 accessed the site on Sept. 23, 2001, at 5:46 a.m. (you can use the Web at any time, you know), going to the library home page (index.html) from the Admissions Test information page (admiss/test.html), using Internet Explorer 5.0 (MSIE 5.0) on a Windows 98 machine.

Job Two: Information Architecture

¶15 The next phase in creating the Web site involves working on the information architecture. A good place to start is deciding on the depth of the structure. Large sites can be broad, with many links on the home page, or they can be deep with only a few links but a structure that users “drill” into. You can draft an initial site map, literally sketching out how the pieces of information will tie together.¹³ The map can be in the form of a flowchart, a spreadsheet—basically, whatever works well for you. Just be sure to use consistent labels, particularly for the menus that will start to emerge from the map. It is best if the main menu, for example, is all nouns or all verbs, not a mixture. It is even better if menus can be broken into types—typically subject and navigation. Similarly, this is the time to decide on file-naming conventions. Is it all right to use a mixture of upper and lower case or not? And what are the conventions for handling regular additions to the site, such as a weekly newsletter? Close on the heels of this decision is one on how to handle archives.

¶16 Tied into these decisions will be the critical choice of what portions of the site, if any, to create dynamically, using database-to-Web systems, such as ColdFusion, PHP/MySQL, or ASP. While setting up such systems requires programmers, the decision itself is one that is fairly natural for librarians. Similarly, librarians are well suited to decide what types of files, if any, to have in Adobe Acrobat’s PDF format. More foreign to us, but also in need of advance planning, is what other “plug-ins” to use in addition to Acrobat. (Plug-ins are pieces of software that users might or might not have attached to their browsers.) This typically means deciding what multimedia will enhance the purposes of the Web site, ranging from Flash animations to audio and video.

¶17 Decisions relating to all of these elements, and more, can be sketched into the draft site map. The final step in this stage is reviewing the draft with the core group and then adapting it as needed.

Job Three: Graphic Arts

¶18 The next phase is designing the site’s look and feel. To do this, you usually create two graphic or design templates. For a small site, one will do, while complex sites will need more. However, two will cover the home page and the basic look of all secondary pages. This process is surprisingly straightforward. First you sketch a basic layout. This can be done on a regular piece of paper in landscape position, which mimics a monitor screen quite nicely. You can draw boxes showing where different chunks of information (header, menus, content, graphics, and footer) will fall. Next you will need to decide how this layout will be held together. This is one of the great peculiarities of the Web. Pages do not hold together like they do in print publishing software. Invisible tables are the glue holding most Web pages

13. See *infra* Appendix B for a sample Web site map.

together. Frames and layers can also be used for layout, although they are not generally considered as good as tables. Cascading style sheets (CSS) are actually preferable to all of these choices, but given inconsistent browser support of CSS, for now it can only be used in a limited fashion.

¶19 Related to this decision is the choice of the page width and placement. Ideally pages will be “liquid,” meaning they can shrink and expand according to monitor size, but in reality, this can conflict with the need to control placement of elements. It will probably have to be decided based on your users. If most have monitors that are 800 pixels wide, that would be the width to use. Aesthetic considerations, including the color scheme and fonts to use, come next. Books and more books can and have been written on colors and typography, so suffice it to say, use an expert or, failing that, good judgment.

¶20 As these pieces fall into place, it is important to watch the total file size (including all the files, such as graphics, associated with the pages). Ideally the total file size will be under 50K. Sizes larger than this do not work well for users with modems connecting to your site.

¶21 Once the basic design templates are done, it is again time to consult with the core group and adapt as needed.

Job Four: Programming

¶22 It is only when this stage is reached that you are ready to embark on the “core” function—developing the basic code of the site. With the groundwork laid, this essentially means translating the design templates developed in the previous stage into code templates.

¶23 Chances are that parts of these design templates (logos, for example) will need to be graphic files. Thus the first step is creating these graphic files for the code to work with. Then, at last, one gets to create some real HTML code. This is also the time to develop the other code (be it CGI, PHP, Javascript, or some other script). If you had two visual templates in the third phase, you will probably need two code templates at this point.

¶24 I cannot tell you how important it is to get the code right at this stage. Errors in these basic templates can become systemic and, as such, very difficult to get rid of. Two things to watch for in particular are (1) using “alt” tags for graphics (so the visually impaired and other users without graphics have a verbal description of the picture they cannot see); and (2) using meta tags. The latter are invisible to the user, but are used—by search engines in particular—both for finding and describing pages. Similarly, now is the time to check for adherence to any guidelines.

¶25 Once you have checked—and double-checked—the code, it is time to move the templates to a “staging site,” which usually means uploading them to a private portion of the Web. From here, you can check to see how the pages look on the browsers and platforms your audience will be using. As a final check, it is good to “validate” your code, using Web resources such as the W3C HTML Validation

Service¹⁴ and Bobby¹⁵ that find errors in the code that you might otherwise overlook.

¶26 The last step (can you guess?) is having the core group review these templates and adapting them as needed.

Job 5: Publishing

¶27 When all of the planning is done, you are ready to do the grunt work—coding all of the pages and broadcasting them. The first step is again one of those surprisingly challenging ones (more likely to be the downfall of a webmaster than HTML will ever be), and that is being sure you have been given all of the content for the site. Then you create all of the HTML pages, graphics, PDF, and other files. When you emerge several weeks later, no doubt with a glazed look in your eye, it is time to proofread everything. This is much like any other proofing, except that it is not linear and you must test all links. Once proofed and corrected, it is time to upload to the server, test all pages on the live site, and then celebrate. But don't stop here (a common mistake). Now that all of this hard work is done and you have a fabulous site, be sure to announce it in newsletters and on electronic discussion lists, and register it with the major search engines. Finally (the ultimate test of your webmaster mettle), be sure to set up a site maintenance schedule and create documentation for the next webmaster. Then, and only then, can you rest on your laurels.

And in the End . . .

¶28 I realize not all readers will rush out and build a Web site from scratch, now or ever, but my intent has been not only to demystify what is involved, but also to break down some of the barriers that seem to prevent librarians from taking a more active role in developing the Web. It is a wild and entertaining ride, full of learning and rewards, and I hope more librarians will soon join the ranks of the Web “wizards.”

14. For more information, see W3C HTML VALIDATION SERVICE, at <http://validator.w3.org/> (last visited Nov. 5, 2001).

15. For more information, see CAST BOBBY, at <http://www.cast.org/bobby/> (last visited Nov. 5, 2001) (Bobby's goal is to “identify and repair significant barriers to access by individuals with disabilities.”).

Appendix A

Checklist for Web Site Development

Job One: Administration—Set the Goals

1. Get the basics in place.
 - a. Determine the core group you will be working with or reporting to;
 - b. See if they have guidelines to be followed;
 - c. Get acquainted with the tools you will be using.
2. Determine the user needs. Possibilities include:
 - a. Work with your core group;
 - b. Work with a Web committee;
 - c. Work with key staff;
 - d. Work with the previous webmaster;
 - e. Do a survey;
 - f. Review log files.
3. Combine the results of one and two in a draft.
4. Get a Web team in place.
 - a. Webmaster
 - b. Information architect
 - c. Art editor / designer
 - d. Photographers
 - e. A/V technician
 - f. Writers
 - g. HTML coders
 - h. Programmers
 - i. Copy editors
5. Develop a schedule.

Job Two: Information Architecture—Blueprint the Structure

1. Decide the depth of the site structure.
2. Create a site map.
3. Figure out consistent labels.
4. Decide on naming conventions.
5. Decide about archives.
6. Decide on programming needs: databases, programming.
7. Decide about PDF.
8. Decide on other plug-in (multimedia) needs: audio, video, Flash.
9. Review blueprint with core group and adapt.

Job Three: Graphic Arts—Design the Look and Feel

1. Sketch the basic layout, starting to draft one or more design templates.
2. Decide how the layout will be held together—frames, tables, layers, CSS.

3. Decide on width and placement.
4. Choose a color scheme.
5. Choose a font set.
6. Monitor total file size.
7. Flesh out these design templates.
8. Review the look with core group and adapt.

Job Four: Programming—Develop the Basic Coding

1. Create graphics files for templates.
2. Turn design templates into code templates of HTML and possibly scripting languages.
3. Use ALT tags for graphics.
4. Use meta tags.
5. Check templates against guidelines (if there are any).
6. Load templates to a “staging site” and review on all the most important platforms.
7. Validate the code.
8. Double-check with core group and adapt.

Job 5: Publishing—Broadcast the Site

1. Make sure the content has been delivered.
2. Do graphics and markup for all pages.
3. Produce PDF and multimedia files.
4. Proofread.
5. Test all pages and links.
6. Upload to the server.
7. Test all pages again.
8. Announce it in electronic discussion lists, e-mail, newsletters, etc.
9. Register it with search engines.
10. Set up a maintenance schedule.
11. Create documentation for next webmaster.

Appendix B Sample Web Site Map

Web Site Map for Merlin, Morgan & Oz, LLP

