Accessibility First! A New Approach to Web Design

Brian J. Rosmaita
Department of Computer Science
Hamilton College
198 College Hill Road
Clinton, NY 13323
brosmait@hamilton.edu

ABSTRACT

This paper proposes an *accessibility first* pedogogy for web design, in which the course is organized around the requirement of implementing web pages accessible to visually impaired computer users. This approach and its advantages are discussed in detail.

Categories and Subject Descriptors

K.3.2 [**Computers and Education**]: Computer and Information Science Education—*computer science education, curriculum*; K.4.2 [**Computers and Society**]: Social Issues—*assistive technologies for persons with disabilities*; I.7.2 [**Document and Text Processing**]: Document Preparation—*markup languages*

General Terms

Human Factors

Keywords

accessibility, CSS, HTML, pedagogy, visually impaired computer users, XHTML, XML

1. INTRODUCTION AND ASSUMPTIONS

In this paper, I present an *accessibility first* approach to teaching web design as part of the computer science curriculum. After a brief discussion of the current standard approach to pedagogy in this area, I'll explain what an accessiblity first approach entails, drawing examples from a course I've designed around the principles articulated herein. The remainder of the paper will consist of a discussion of the advantages of teaching web design in this way.

For the purposes of this paper, discussion is limited to a course in web design observing the following three assumptions.

1. The course is introductory-level (i.e., no previous computer science experience is assumed).
2. The course is not major-specific (i.e., the course is open to non-majors and is not expected to satisfy the demands of an introductory course for majors).
3. The course has room for some discussion of social aspects of computing.

(Note that in practice, courses meeting the first two assumptions almost always meet the third assumption.)

One other assumption I should make clear is that this *is* a computer science course. I mention this because many computer scientists feel that web design is not an appropriate topic of instruction for a Computer Science Department. Reasons I've heard for this are that markup languages are too easy, markup languages aren't programming languages, or that teaching web design is akin to teaching students how to use a word processor. As these same thoughts may have occurred to the reader, I'd like to explain why they're misguided.

Markup languages are too easy: I interpret this concern to be that markup languages are too easy to be taught *in a formal course setting*. (Even a casual look at the World Wide Web reveals that markup languages aren't so easy that everyone uses them correctly). At the introductory level, however, I don't see that this is the case at all—particularly if students are required to write formally valid documents. Writing valid XHTML, for example, requires attention to case-sensitivity, proper closing of tags, proper nesting of tags, proper use of entity codes, etc. Running the document through a markup language validator is similar to compiling a program: students receive cryptic error messages often referencing lines of code that don't themselves contain the problem, and students must have a good knowledge of the formal structure of the language to decipher these messages and make corrections. Now while a student who's learned a markup language isn't thereby ready for CS2, this course isn't meant to be a substitute for CS1. On the other hand, a student who's been through a course like this is well-prepared for CS1—and the student has a pretty good idea of what's to come. (If you don't like figuring out how to get your web pages to validate, you're going to *hate* CS1!)

Markup languages aren't programming languages: This statement is certainly true, but it's really not to the point. It's easy to include programming in a web design course: one could teach JavaScript, PHP, or XML and XSLT (the latter being a declarative programming language for transforming XML documents into, e.g., HTML documents.) Alternatively, one could use Emacs as the web authoring tool, teach the students Elisp, and have them write code to en-

SIGCSE'06 March 1–5, 2006, Houston, Texas, USA.

rich the authoring tool. More fundamentally, however, one could simply take issue with this criticism's unstated assumption that computer science is really about programming, and leave it at that.

Teaching web design is akin to teaching students how to use a word processor: It *is* possible to teach web design that way—as simply learning the ins and outs of a particular authoring tool—but this is not *necessarily* the case. Further, if one is interested in interoperability and accessibility, it's *impossible* to simply teach an authoring tool without also teaching the target markup language, since one often has to tweak the generated code to achieve these goals. Finally, if a goal of the course is to teach students the competent use of a formal language, you're not going to use a very powerful authoring tool anyway—you'll want students to construct their web pages "by hand". So this criticism is misguided as well.

To summarize: it is certainly possible to teach a "real" computer science course on this topic, particularly under the three assumptions set out above.

One final assumption. In speaking of 'accessibility', the primary concern in this paper is accessibility for the visually impaired. This fits in with the current practice of teaching web design as a type of Graphical User Interface design. This is not to denigrate the importance of other types of accessibility; it just reflects the fact that current websites are designed to be used by browsers operating under a point-and-click paradigm.

2. IMPORTANCE OF ACCESSIBILITY

Since teaching accessibility *first* implies that accessibility should be taught at all, it is worth examining why accessibility is a topic of interest to computer scientists.

The Social Responsibility Argument: The World Wide Web Consortium (W3C) was founded by Tim Berners-Lee in 1994. Its purpose is to "lead the World Wide Web to its full potential by developing common protocols that promote its evolution and ensure its interoperability" [14]. In practice, this means issuing technical specifications for the Web's infrastructure.

Of particular importance here is what the W3C identifies as the *first* of its long-term goals for the web:

> *Universal Access:* To make the Web accessible to all by promoting technologies that take into account the vast differences in culture, languages, education, ability, material resources, access devices, and physical limitations of users on all continents [14].

By designing web pages accessible to the visually impaired, we clearly make a contribution to universal access.

The Legal Argument: Section 508 of the Rehabilitation Act [13] (as amended in 1998) requires that Federal agencies' electronic and information technology be accessible to people with disabilities, where this latter class includes both federal employees and members of the public.

Some states have adopted similar measures. For example, New York Statewide Technology Policy P04-002 [8], requires that web content provided by state agencies be accessible to persons with disabilities.

There is, then, a large number of websites that must be accessible as a matter of law; this makes accessibility a topic of interest to anyone who hopes to work with such websites.

Some people don't find the social responsibility argument for accessiblity to be very convincing: it establishes that *someone* should promote accessiblity, such a person might say, but it doesn't explain *why this someone is me*, and hence it doesn't provide *me* with a reason for implementing accessible web content. Similarly, the legal argument only impacts upon those who wish to work with particular websites to which the laws apply, and hence does not make accessibility a general concern. We need another way to think about this issue in order to extend its range of interest. Fortunately, we don't have to look very far.

The March of Technology Argument: As technology develops, there are more and varied ways to access the content of the World Wide Web. For example, people may wish to surf the web while driving an automobile using voice commands and auditory response. So automobile drivers—who otherwise have normal vision—are blind with respect to the web while they are driving. Likewise, a person surfing the web on a small mobile handheld device is, for all intents and purposes, a low-vision person accessing the web.

In short, the march of progress is inevitably making accessibility a concern not simply for those who wish to reach that small share of the market represented by the physically disabled, but also for those who will wish to reach the majority of computer users. So while there is certainly a moral argument for making web content accessible, and a legal argument for making at least some web content accessible, it's interesting to note that there is a backup argument for accessibility based on self-interest, completely independent of any legal or moral concerns.

Given the discussion in this section, it is clear that accessiblity is a topic of interest to computer scientists and computer science educators.

3. CURRENT PRACTICE

The standard approach to teaching introductory web design is to instruct students in the HyperText Markup Language (HTML), usually with exposure to a scripting language (e.g., JavaScript) for implementing interactivity. Control of style is through the style attributes of HTML tags and (perhaps) some use of style sheets. Most current textbooks include a discussion of webpage interoperability and accessibility, though the emphasis on this varies greatly. Accessibility is most often addressed through the use of the ALT attribute for IMG elements (ALT is a required attribute for IMG in HTML 4.01 [16], so most textbooks feel obligated to mention it).

This dearth of discussion of accessibility is borne out by a survey of some popular textbooks. The fifth edition of Musciano and Kennedy's 645 page *Definitive Guide* [7], for example, does not contain an entry for 'accessibility' in the index. Likewise, the index of Crowder and Bailey's 930 page *Creating Web Sites Bible* is notable for the absence of any mention of 'accessibility'.

Elizabeth Castro's *Visual Quickstart Guide* [2] (whose cover proclaims it "the #1 best-selling book on HTML") contains three pages (out of 480) on which accessibility is mentioned. (Two of these are simply statements that US Government websites are required by law to be accessible; the third occurrence is a little more substantive, mentioning the use of the LONGDESC attribute for frames.) Dan Cederholm's *Bulletproof Web Design* [3] contains a two-page discussion (out of 270 pages) of accessiblity which occurs

about two-thirds of the way through the text; the discussion is focused solely on table design.

On a brighter note, chapter 6 of Jennifer Niederst's *Web Design in a Nutshell* [9] is titled "Accessibility". This chapter contains a summary of the Web Content Accessibility Guidelines [19], but is only seven pages of a 618 page text. Further, the index entry for 'accessibility' refers only to pages 74-80, i.e., all of chapter 6—the discussion of accessibility is isolated from the discussion of actual design and coding techniques. (Niederst's *Learning Web Design* [10], by the way, doesn't contain an entry for 'accessibility' in the index.) The "Comprehensive" edition of Patrick Carey's *New Perspectives on HTML and XHTML* [1] contains a section on "Making the Web More Accessible"; this, however, is relegated to Appendix D of the 736 page text.

In general, in the standard approach, when accessibility is mentioned at all, it is treated as an "add-on" (like ALT text for HTML IMG elements)—something extra that one can do, but ancillary to the primary task of designing visually attractive web pages. As a result, many students treat ALT text like comments in CS1—something you go back and throw in at the end, after you've done the important stuff. Further, some of the requirements for accessibility conflict with common web design techniques (e.g., use of tables for page layout), so students receive a mixed message about the true importance of accessibility.

The social aspects of computing discussed in a standard course on this topic—following the available textbooks—usually focus on privacy concerns, e-business, and hacking. Accessibility does not receive much attention.

4. THE NEW APPROACH

Instead of teaching accessibility as an "add-on", I propose that it be brought front and center and made the focus of the course. All aspects of web design should be taught from the standpoint of how they contribute to accessibility. This may seem odd, since the visually impaired are clearly a minority of those viewing web pages, but as we'll see, there are many advantages to this approach.

4.1 Course Coverage

Here's a quick outline of the topics that would be covered in such a course.

1. Informal discussion of accessible web design
2. Instruction on the authoring tools to be used, basic HTML, mounting a website on the server
3. Readings from disability studies literature on blindness
4. Instruction in XHTML and CSS
5. Formal discussion of accessible web design and general usability principles
6. Discussion of social aspects of computing via Section 508 of the Rehabilitation Act

(Note that the outline is only roughly temporal; items 2 and 3 should occur in parallel, as should items 3, 4, and 5.)

I'll now go back and fill in the outline, drawing examples from a recent course I taught along these lines [12]. Keep in mind that there are many ways to implement this new approach; the specific choices I made are not meant to be definitive—just examples of how it can be done.

Informal discussion of accessible web design: Unless they are personally acquainted with a visually impaired computer user, most students are surprised to learn that there are blind people who surf the web. After a few minutes of thought, however, the surprise dissipates: *everyone* wants to use the web, so it isn't surprising that blind people would want to, also; most students just haven't given the matter any thought previously. So at the first class meeting, I explain to the students that we're going to learn how to implement websites so that both sighted and visually impaired persons will equally be able to access their content. (This is where teaching an introductory class is an advantage—as far as my students are concerned, this is the natural way to approach web design.)

A nice feature of HTML is that it's intuitive enough that even on the first day of class, students can be shown examples of marked-up text that display identically in a browser, yet which, when the "raw" HTML is examined, are clearly distinct, with one being clearly preferable from a purely syntactic standpoint. An example the students grasp quickly is the following. Consider these two fragments of markup that display identically in a web browser:[1]

```
<p>&bull;  Vinton Cerf</p>
<p>&bull;  Tim Berners-Lee</p>
<p>&bull;  Al Gore</p>
```

```
<ul>
<li>Vinton Cerf</li>
<li>Tim Berners-Lee</li>
<li>Al Gore</li>
</ul>
```

Once they're told how to interpret the tags, students immediately understand that there's a clear difference between the two examples and that the latter way of marking up a list is better—placing the names in different, though contiguous, paragraphs does not preserve their relationship of being items in the *same* list. Further, it's easy to see that a computer could parse the second example to determine that 'Al Gore' is the third item of a three-item list, but it's not obvious how to parse the paragraph markup to determine *that* it's a list nor how many items it contains.

After looking at this toy example, students can be shown real webpages illustrating the same problem (they aren't difficult to find). This little exercise is very empowering for students: they learn, on the very first day of class, both how to look at HTML code in a browser and how to read HTML to the extent that they can distinguish good markup from poor markup. Further, they notice that the true list markup is cleaner and easier to read, even in HTML. So, if you're going to be typing this stuff in, they figure, why not do it correctly, especially when it will make the content accessible to a wider audience?

Another way to bring accessibility to the forefront of the course is to have students surf the web using screen reading software and the monitor turned off. They will very quickly wander into frustratingly inaccessible regions of the web. This gives students a motivation, based in personal experience, for marking up text in an accessible way.

Instruction on the authoring tools to be used, basic HTML, mounting a website on the server: This is pretty straightforward. I prefer to use open-source software for this course, so

[1]To get *identical* display, of course, the style properties of the paragraph element have to be modified from the default, but this isn't a detail students need to know about on the first day of class.

students learn to use command-line GNU/Linux with Emacs as an authoring tool. To practice Emacs and make sure they understand file permissions, etc., students construct a very simple webpage to post on the server immediately.

Readings from disability studies literature on blindness: Accessible design takes more effort, so it's good for students to have some personal motivation. I've used Rod Michalko's *The Two-in-One* [6], a memoir of a blind sociology professor, for this purpose. To prepare for discussion, students are required to post a webpage of chapter summaries and commentary, which also provides me with a source of web design exercises. In connection with this, a class visit from a blind computer user is extremely effective, particularly if you can find someone of college age.

Instruction in XHTML and CSS: The markup language I use is XHTML 1.0 [20], with CSS 2 [15] to control style. I've used Jon Duckett's text [4], though mostly as a reference guide and not a textbook. (He's sensitive to accessiblity issues, though he doesn't take the approach I'm advocating here.[2]) Students are expected to write valid XHTML and CSS, as verified by an appropriate validator [18, 17].

Formal discussion of accessible web design and general usability principles: For the purposes of this course, web accessibility is defined as conformance with the Web Content Accessibility Guidelines (WCAG 1.0) [19]. For general usability principles, I rely on Jakob Nielsen's "Alertbox" columns, posted on his website [11]. They're free, short, well-written, very informative, and students like reading them.

Discussion of social aspects of computing via Section 508 of the Rehabilitation Act: The more formal social aspects of computing materials focus on Section 508 of the Rehabilitation Act (as amended in 1998) [13]. Section 508 requires that Federal agencies' electronic and information technology be accessible to people with disabilities. The law is not binding on private citizens. Further, it is not clear that the provisions for enforcement are in any way effective. (Basically, each Federal agency assesses its own compliance with the law.)

Section 508 raises many interesting issues. For example, does it mandate too much? Not enough? If it's good for Federal agencies, should a similar policy be extended to state agencies? What about to the private sector?

4.2 Advantages of the New Approach

Having outlined the new approach, let's take a look at some of its advantages.

Clear Separation of Content and Presentation: In order to be in compliance with the WCAG 1.0, an author must clearly separate content from presentation. For example, a book title (*Inroads*) and emphasized text (webpages *must* be accessible) are both traditionally displayed in italics, i.e., they have the same presentation. But they are different ontologically, since they are different types of things—one is a title, and one is an emphasized word. The two words could both be marked up using an `<i>` tag, which would italicize them both, but a better solution would be to respect their difference and mark the former with a `<cite>` tag and the latter with `<em>`. This difference in markup reflects the difference in content.

Placing an emphasis on accessibility provides students with a good reason to separate content from presentation. It's easy to imagine screen reading software increasing loudness when reading text marked with an `<em>` tag, but it's not so clear what you'd want it to do when encountering a new `<font>`. (Does the font change signify change in emphasis? Does it signify a book title? Does it signify a heading? Does it have no logical significance in the document?) With a strong content/presentation distinction in hand, it is much easier to teach students to use a markup language to mark up the logical structure of a document.

The use of XHTML further reinforces this distinction because strict XHTML lacks the stylistic elements and attributes of HTML. The presentation aspects of a document are moved to its associated style sheet.

XHTML is simply a reformulation of HTML 4.01 in XML, so a further advantage to this approach is that students are learning an XML application. Finally, since XML is much more strict than HTML, it will be easy for students with a knowledge of XHTML to write (or maintain) HTML documents.

Extensibility and the Multi-Modal Web: Creating accessible web content isn't just nice for visually impaired people—it is an important skill for a web author to have as people increasingly surf the web via different modalities. (Predicting the future is always risky, but it's pretty safe to say that people surfing the web while driving will not be pointing and clicking on a monitor!) So this approach to teaching web design will become increasingly important as technology develops.

Another way in which this approach is extensible is in preparing students to write XML documents. The class attributes of XHTML elements can be used to group them in various ways (e.g., different kinds of emphasis). These different kinds of emphasis could be given different XML tags; conversely, one could write an XSLT application to translate these tags into HTML `<em>` elements with different class attributes, tied to a stylesheet specifying a different mode of display for each class.

Student Motivation for Implementing Interoperability: While most current introductory web design texts make a nod toward interoperability, it's difficult for students to see why it should matter if a web page doesn't display in, e.g., Mozilla Firefox—after all, *everyone* uses [insert your favorite browser here]. Accessibility, however, implies interoperability, so if we can convince students to write accessible pages, we get interoperability for free.

Tight Integration of Computer Science and Social Aspects of Computing: While it's interesting to discuss Denial of Service attacks, most of our students will never launch one (thank goodness!). And while a DOS attack may cost the U.S. economy billions of dollars, the most likely impact on an individual student is some minor inconvenience until the attack is brought under control.

In the area of assistive technology, however, it's easy for students to walk in someone else's shoes simply by turning off the computer monitor and trying to navigate the web with screen reading software. It makes the debate over the extent to which websites should be accessible have much more personal impact.

This course must contain a large component of social aspects of computing in order to work—students really need to appreciate the social importance of accessibility in order to be motivated to implement it in their web design. A nice

[2]Duckett is also the author of *Accessible XHTML and CSS Web Sites* [5]. That book, however, assumes prior knowledge of HTML and website construction.

feature of the accessibility first approach is that the social aspects are integrated right into the "normal" stuff you'd do anyway in a course of this nature.

Service Learning: Accessibility first web design provides an opportunity to include service learning in an introductory level class. Students can be given the assignment of creating an accessible "shadow" website for a campus organization, preserving as much of the functionality of the original website as possible. This is not too ambitious a goal—the websites of most campus organizations are either very primitive (easy to fix) or generated by an authoring tool (incredibly inaccessible). So virtually anything a competent student does will be an improvement. (Further, since the student is building a shadow website, the organization doesn't have to accept the changes.)

A nice aspect of including a service learning component is that students can see that what they're doing in class can have an immediate impact out in the world. Further, they will get a real taste of how difficult it is to retrofit accessibility—it's much better to start out with an accessible design from the beginning.

This is yet another reason to put accessibility first. It's no fun writing accessible web pages if you have to go back and unlearn everything in order to make your pages accessible. If you keep accessiblity in mind from the beginning, it's much easier to implement.

Aging Baby Boomers: Visually impaired computer users are a minority, but it's a growing minority, and it is growing faster as baby boomers near retirement age. Further, it's a minority that will eventually include us all.

It's a Good Thing! Finally, I think that all would agree that when faced with an inaccessible website and an accessible website with the same functionality, the accessible website is better. The debate is really over *who pays* to implement accessibility, and *why* they should have to bear that cost.

As I indicated above, as it becomes more common for people to surf the web via different modalities, accessibility will become a necessity for content providers. So I think the cost argument will be settled in the usual way—providers will pay because it's in their own self interest.

As far as personal web pages go, a person who (a) thinks that accessiblity is a good idea and (b) knows how to implement accessibility, is going to design accessible web pages. Students taking this course will meet both criteria, so an advantage of this way of teaching is that the quality of web pages (at least from an accessibility standpoint) will go up.

5. CONCLUSION

Placing the issue of creating web content accessible to visually impaired computer users at the forefront of a course on web design both stresses the importance of good web design and provides students with the motivation to implement good web design.

- It gives students a *concrete practical reason* for respecting the content/presentation distinction.
- It gives students a *practical model* for making the content/presentation distinction when faced with marking up a document.
- It results in web pages accessible to visually impaired computer users.

Everybody wins.

6. REFERENCES

[1] P. Carey. *New Perspectives on HTML and XHTML, Comprehensive*. Thomson Course Technology, Boston, MA, 2004.

[2] E. Castro. *HTML for the World Wide Web with XHTML & CSS: Visual Quickstart Guide*. Peachpit Press, Berkeley, CA, 2003.

[3] D. Cederholm. *Bulletproof Web Design: Improving Flexibility and Protecting Against Worst-Case Scenarios with XHTML and CSS*. New Riders, Berkeley, CA, 2005.

[4] J. Duckett. *Beginning Web Programming with HTML, XHTML, and CSS*. Wrox, an imprint of Wiley Publishing, Inc., Indianapolis, IN, 2004.

[5] J. Duckett. *Accessible XHTML and CSS Web Sites: Problem-Design-Solution*. Wrox, an imprint of Wiley Publishing, Inc., Indianapolis, IN, 2005.

[6] R. Michalko. *The Two-in-One: Walking with Smokie, Walking with Blindness*. Temple University Press, Philadelphia, 1999.

[7] C. Musciano and B. Kennedy. *HTML and XHTML: The Definitive Guide*. O'Reilly Media, Inc., Sebastapol, CA, fifth edition, 2002.

[8] New York State Office for Technology. *Accessibility of State Agency Web-based Intranet and Internet Information and Applications* (New York Statewide Technology Policy P04-002). `http://www.oft.state.ny.us/policy/p04-002/`.

[9] J. Niederst. *Web Design in a Nutshell: A Desktop Quick Reference*. O'Reilly Media, Inc., Sebastapol, CA, second edition, 2001.

[10] J. Niederst. *Learning Web Design: A Beginner's Guide to HTML, Graphics & Beyond*. O'Reilly Digital Studio. O'Reilly Media, Inc., Sebastapol, CA, second edition, 2003.

[11] J. Nielsen. useit.com: Jakob Nielsen on Usability and Web Design. `http://www.useit.com/`.

[12] B. Rosmaita. The CPSCI 107 Homepage. `http://warp.cs.hamilton.edu/course.html`.

[13] U.S. Department of Justice. Section 508 Home Page. `http://www.usdoj.gov/crt/508/`.

[14] World Wide Web Consortium. About the World Wide Web Consortium (W3C). `http://www.w3.org/Consortium/`.

[15] World Wide Web Consortium. Cascading Style Sheets, level 2 (CSS2 Specification). `http://www.w3.org/TR/1998/REC-CSS2-19980512/`.

[16] World Wide Web Consortium. HTML 4.01 Specification. `http://www.w3.org/TR/1999/REC-html401-19991224/`.

[17] World Wide Web Consortium. W3C CSS Validation Service. `http://jigsaw.w3.org/css-validator/`.

[18] World Wide Web Consortium. W3C Markup Validation Service. `http://validator.w3.org/`.

[19] World Wide Web Consortium. Web Content Accessibility Guidelines 1.0. `http://www.w3.org/TR/1999/WAI-WEBCONTENT-19990505/`.

[20] World Wide Web Consortium. XHTML 1.0: The Extensible HyperText Markup Language (Second Edition). `http://www.w3.org/TR/2002/REC-xhtml1-20020801/`.