chapter

11

Accessibility, Cascading Style Sheets, and you

just when you think you're done, a cat floats by with buttered toast strapped to its back

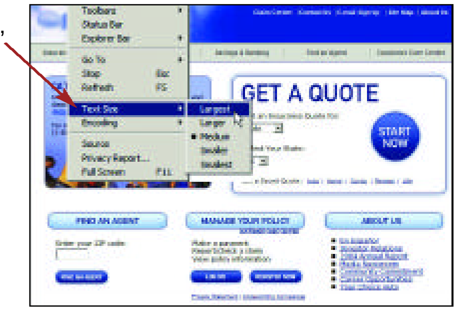
> When a cat is dropped, it always lands on its feet, and when toast is dropped, it always lands with the buttered side facing down. I propose to strap buttered toast to the back of a cat; the two will hover, spinning, inches above the ground. With a giant buttered-cat array, a high-speed monorail could easily link New York with Chicago.
>
> —john frazee, in *the journal of irreproducible results*

People sometimes ask me, "What about accessibility? Isn't that part of usability?"

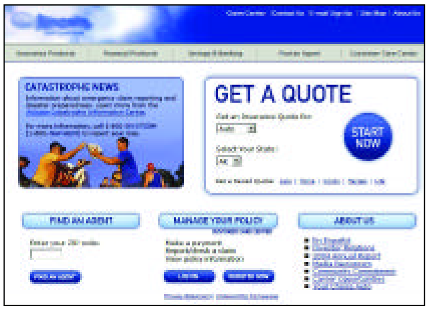
And they're right, of course. Unless you're going to make a blanket decision that people with disabilities aren't part of your audience, you really can't say your site is usable unless it's accessible.

At this point,[1] everyone involved in Web design knows at least a little bit about Web accessibility, even if it's only that there's something special about the number 508.[2] And yet almost every site I go to fails my three-second accessibility test—increasing the size of the type.

Browser "Text Size" command



Before



After (no difference)

Why is that?

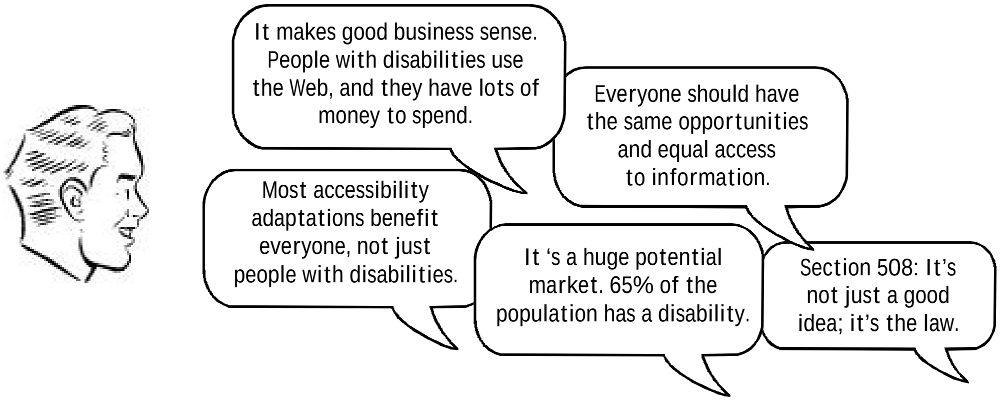
[1] 2005 AD

[2] In case you've literally been hiding under a rock for the past few years, "508" refers to Section 508 of the 1988 Amendments to the Rehabilitation Act, which specifies accessibility standards for information technology (like Web sites) that must be met by any vendor that wants to do business with the U.S. government.

What developers and designers hear

In most organizations, the people who end up being responsible for doing something about accessibility are the people who actually build the thing: the designers and the developers.

When they try to learn about what they should do, whatever books or articles they pick up inevitably list the same set of reasons why they need to make their sites accessible:


It makes good business sense. People with disabilities use the Web, and they have lots of money to spend.
Everyone should have the same opportunities and equal access to information.
Most accessibility adaptations benefit everyone, not just people with disabilities.
It ‘s a huge potential market. 65% of the population has a disability.
Section 508: It’s not just a good idea; it’s the law.

There’s a lot of truth in all of these. Unfortunately, there’s also a lot that’s unlikely to convince 26-year-old developers and designers that they should be “doing accessibility.” Two arguments in particular tend to make them skeptical:

- Since their world consists largely of able-bodied 26-year-olds, it’s very hard for them to believe that a large percentage of the population actually needs help accessing the Web. They’re willing to write it off as the kind of exaggeration that people make when they’re advocating for a worthy cause, but there’s also a natural inclination to think, “If I can poke a hole in one of their arguments, I’m entitled to be skeptical about the rest.”
- They’re also skeptical about the idea that making things more accessible benefits everyone. Some adaptations do, like the classic example, closed captioning, which does often come in handy for people who can hear.[3] But since this always seems to be the only example cited, it feels a little like arguing

[3] Melanie and I often use it when watching British films, for instance.

that the space program was worthwhile because it gave us Tang.[4] It's much easier for developers and designers to imagine cases where accessibility adaptations are likely to make things *worse* for "everyone else."

The worst thing about this skepticism is that it obscures the fact that there's really only one reason that's important:

> It's the right thing to do.

And not just the right thing; it's *profoundly* the right thing to do, because the one argument for accessibility that doesn't get made nearly often enough is how extraordinarily better it makes some people's lives. Personally, I don't think anyone should need more than this one example: Blind people with access to a computer can now read the daily newspaper on their own. Imagine that.

How many opportunities do we have to dramatically improve people's lives just by doing our job a little better?

And for those of you who don't find this argument compelling, be aware that there will be a legislative stick coming sooner or later. Count on it.

What designers and developers fear

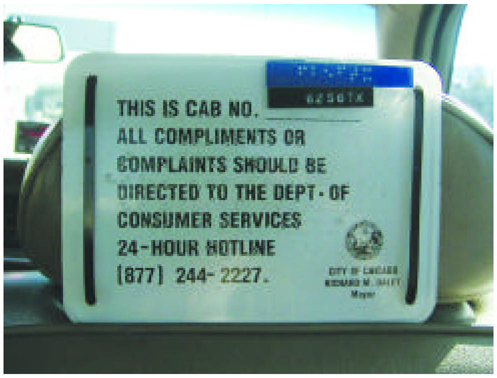
As they learn more about accessibility, two fears tend to emerge:

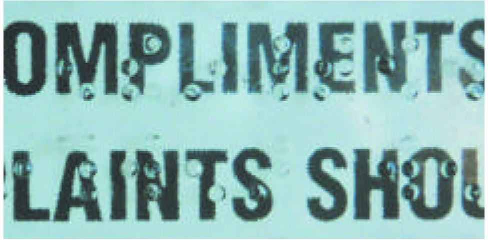
> **More work.** For developers in particular, accessibility can seem like just one more complicated new thing to fit into an already impossible project schedule. In the worst case, it gets handed down as an "initiative" from above, complete with time-consuming reports, reviews, and task force meetings.

> **Compromised design.** What designers fear most is what I refer to as buttered cats: places where good design for people with disabilities and good design for everyone else are going to be in direct opposition. They're worried that they're going to be forced to design sites that are less appealing—and less useful—for the majority of their audience.

[4] A powdered orange-flavored breakfast drink, invented for the astronauts (see also: freeze-dried food).

In an ideal world, accessibility would work like a sign I saw in the back of a Chicago taxi. At first it looked like an ordinary sign. But something about the way it caught the light made me take a closer look, and when I did, I realized that it was ingenious.





The sign was overlaid with a thin piece of Plexiglas, and the message was embossed in Braille on the Plexiglas. Ordinarily, both the print and the Braille would have been half as large so they could both fit on the sign, but with this design each audience got the best possible experience. It was an elegant solution.

I think for some designers, though, accessibility conjures up an image something like the Vonnegut short story where the government creates equality by handicapping everyone.[5]

The real solution—as usual—is a few years away

When people start reading about accessibility, they usually come across one piece of advice that sounds very promising:

[5] In "Harrison Bergeron," the main character, whose intelligence is "way above normal," is required by law to wear a "mental handicap radio" in his ear that blasts various loud noises every 20 seconds "to keep people like George from taking unfair advantage of their brains."

Use a validator like Bobby to make sure your site complies with the WCAG guidelines.
Great! A spell checker for accessibility.

The problem is, when they run their site through a validator, it turns out to be more like a grammar checker than a spell checker. Yes, it does find some obvious mistakes and oversights that are easy to fix, like missing alt text.[6] But it also inevitably turns up a series of vague warnings that you may be doing something wrong, and a long list of recommendations of things for you to check which it admits may not be problems at all.

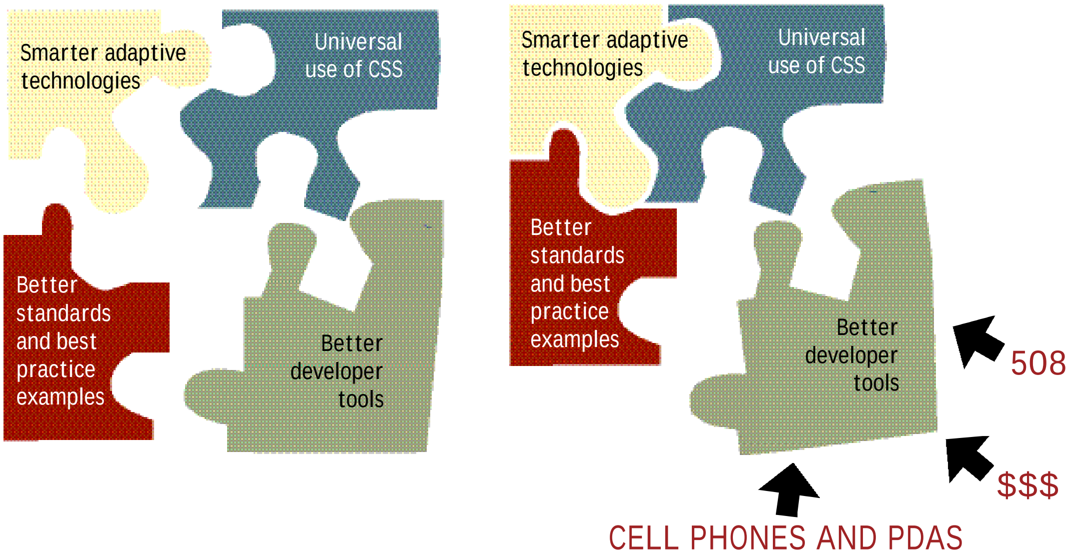
This can be very discouraging for people who are just learning about accessibility, because the long lists and ambiguous advice suggest that there's an awful lot to learn.

And the truth is, it's a lot harder than it ought to be to make a site accessible.

After all, most designers and developers are not going to become accessibility experts. If Web accessibility is going to become ubiquitous, it's going to have to be easier to do. Screen readers and other adaptive technologies have to get smarter, the tools for building sites (like Macromedia Dreamweaver) have to make it easier to code correctly for accessibility, and the guidelines need to be improved.

[6] Alt text provides a text description of an image ("Picture of two men on a sailboat," for example), which is essential for people using screen readers or browsing with images turned off.

Real progress is going to require improvements on four different fronts, motivated by things like profit incentive, the threat of lawsuits and legislation, and the desire to support mobile devices, which share some problems with accessibility.


Smarter adaptive technologies
Universal use of CSS
Better standards and best practice examples
Better developer tools
Smarter adaptive technologies
Universal use of CSS
Better standards and best practice examples
Better developer tools
508
$$$
CELL PHONES AND PDAS

The five things you can do right now

The fact that it's not a perfect world at the moment doesn't let any of us off the hook, though.

Even with current technology and standards, it's possible to make any site very accessible without an awful lot of effort by focusing on a few things that will have the most impact. And they don't involve getting anywhere near a buttered cat.

#1. Fix the usability problems that confuse everyone

One of the things that I find annoying about the Tang argument ("making sites accessible makes them more usable for everyone") is that it obscures the fact that the reverse actually is true: Making sites more usable for "the rest of us" is one of the most effective ways to make them more effective for people with disabilities.

If something confuses most people who use your site, it's almost certain to confuse users who have accessibility issues. (They don't suddenly become remarkably smarter because they have a disability.) And it's very likely that they're going to have a harder time recovering from their confusion.

For instance, think of the last time you had trouble using a Web site (running into a confusing error message when you submitted a form, for instance). Now imagine trying to solve that problem without being able to see the page.

The single best thing you can do to improve your site's accessibility is to test it often, and continually smooth out the parts that confuse everyone. In fact, if you don't do this first, no matter how rigorously you apply accessibility guidelines, people with disabilities still won't be able to use it. If your site's not clear to begin with, making it Bobby-compliant is like [insert your favorite putting-lipstick-on-a-pig metaphor here].

#2. Read an article

As I hope you've seen by now, the best way to learn how to make anything more usable is to watch people actually try to use it. But most of us have no experience at using adaptive technology, let alone watching other people use it.

If you had the time and the motivation, I'd highly recommend locating one or two blind Web users and spending a few hours with them observing how they actually use their screen reader software.

Fortunately, someone has done the heavy lifting for you. Mary Theofanos and Janice (Ginny) Redish watched 16 blind users using screen readers to do a number of tasks on a variety of sites and reported what they observed in an article titled "Guidelines for Accessible and Usable Web Sites: Observing Users Who Work with Screen Readers."[7]

As with any kind of user testing, it produced invaluable insights. Here's one example of the kinds of things they learned:

[7] Published in the ACM Magazine, Interactions (November-December 2003). With permission from ACM, Ginny has made it available for personal use at http://redish.net/content/papers/interactions.html.

> Screen-reader users scan with their ears. Most blind users are just as impatient as most sighted users. They want to get the information they need as quickly as possible. They do not listen to every word on the page—just as sighted users do not read every word. They “scan with their ears,” listening to just enough to decide whether to listen further. Many set the voice to speak at an amazingly rapid rate.
>
> They listen to the first few words of a link or line of text. If it does not seem relevant, they move quickly to the next link, next line, next heading, next paragraph. Where a sighted user might find a keyword by scanning over the entire page, a blind user may not hear that keyword if it is not at the beginning of a link or a line of text.

I highly recommend that you read this article before you read anything else about accessibility. In 20 minutes, it will give you an appreciation for the problems you’re trying to solve that you won’t get from any other articles or books.

#3. Read a book

After you’ve read Ginny and Mary’s article, you’re ready to spend a day (or a weekend) reading a book about Web accessibility. There are several good ones...

- *Building Accessible Websites* by Joe Clark
- *Constructing Accessible Websites* by Jim Thatcher et al.
- *Maximum Accessibility: Making Your Web Site More Usable for Everyone* by John Slatin and Sharron Rush
- A CD-ROM called *The WebAIM Guide to Web Accessibility Techniques and Concepts*

...and I’m sure there will be more in the near future.[8]

These books cover a lot of ground, so don’t worry about absorbing all of it. For now, you just need to get the big picture.

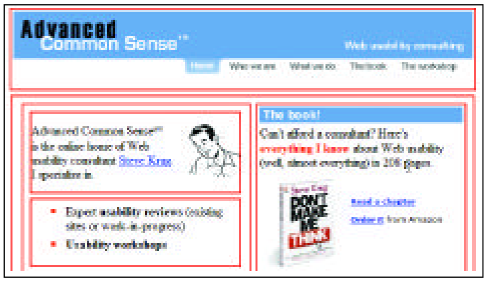
[8] I’ll keep an updated list of recommendations on my Web site.

#4. Start using Cascading Style Sheets

First, a little Web history.

In the beginning, everything was text. When the first visual browsers arrived, designers found that unlike desktop publishing, which gave them control of everything, HTML—which was really only intended to display research papers—gave them almost no control over anything. Commands for styling text were crude, and it was very hard to position things precisely on a page. And even if you could, pages often looked completely different when viewed in different browsers.

To wrestle back some control, designers and developers started using tables to control layout. For years, the only way to control the position of things on a Web page was to put them in tables... and tables within tables. Most pages ended up seeming like a series of Russian nesting dolls.

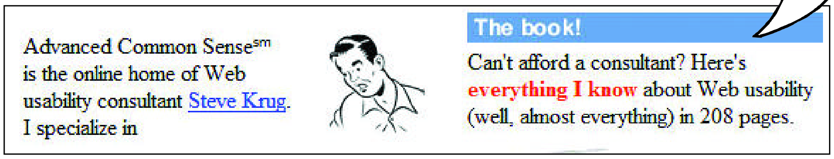


Advanced Common Sense℠ is the online home of Web usability consultant Steve Krug. I specialize in

- Expert usability reviews (existing sites or work-in-progress)
- Usability workshops

Can't afford a consultant? Here's everything I know about Web usability (well, almost everything) in 208 pages.

Unfortunately, this didn't work well with early screen readers, which tended to read rather slavishly line-by-line from left to right, like this:


Advanced Common Sense can't afford a consultant here's is the online home of web everything I know about web usability...
The book!

Advanced Common Sense℠ is the online home of Web usability consultant Steve Krug. I specialize in

Can't afford a consultant? Here's everything I know about Web usability (well, almost everything) in 208 pages.

They also started using various HTML commands in ways they weren't meant to be used, to try to get more control over text styling. It was a messy world of hacks, held together with chewing gum.

Fortunately, beginning in 1998 some very determined people got fed up with this state of affairs, and decided to convince browser manufacturers to support Web standards that would give designers a consistent target. A group of designers calling themselves The Web Standards Project employed a brilliant form of nonviolent resistance: They simply stopped making their own sites backwardly compatible with browsers that didn't support standards like CSS, and encouraged others to do the same.

Several year later, CSS Zen Garden[9] (a single HTML page that looked completely different depending on which of the many designer-contributed style sheets you applied to it) demonstrated that you could create beautiful, sophisticated designs with CSS.

Cascading Style Sheets are now so well supported by most browsers that it doesn't make any sense to create a site without them, because the advantages are enormous:

> Infinitely greater control of formatting.

> Flexibility. A single change in a style sheet can change the appearance of an entire site, or automatically generate useful variations like printer-friendly pages.

> Consistency among browsers. Workarounds and hacks are still required to ensure that your CSS works across all browers, but these will fall away as brower makers continue to improve their CSS support.

And implementing CSS will make it easy for you to do two things that will greatly improve your site's accessibility:

> Serialize your content. Unlike table-based layout, with CSS you can put your content in sequential order in the source file—which is how a screen reader user will hear it—and still position things where you want them on the page.

> Allow your text to resize. CSS makes it easy to make your text resizable, which is enormously helpful for low-vision users (and people old enough to need bifocals).

[9] www.csszengarden.com. See The Zen of CSS Design by Dave Shea and Molly Holzschlag (New Riders, 2005) for a great description of the project.

Probably the fastest way to learn CSS is to get someone who specializes in it to do a "markover" for you—recoding a few of your site's page templates to use CSS—and learn by watching them do it. When you're ready, there are also a number of good books on CSS, especially the ones by Eric Meyer.

#5. Go for the low-hanging fruit

Now you're ready to do what most people think of as Web accessibility: implementing specific changes in your HTML code.

As of right now, these are probably the most important things to do:

> Add appropriate alt text to every image. Add an alt attribute for images that screen readers should ignore, and add helpful, descriptive text for the rest. All of the Web accessibility books have very good explanations of how to do this.

> Make your forms work with screen readers. This largely boils down to using the HTML label element to associate the fields with their prompts, so people know what they're supposed to enter.

> Create a "Skip to Main Content" link at the beginning of each page. Imagine having to spend 20 seconds (or a minute, or two) listening to the global navigation at the top of every page before you could look at the content, and you'll understand why this is important.

> Make all content accessible by keyboard. Remember, not everyone can use a mouse.

> Don't use JavaScript without a good reason. Some adaptive technologies don't support it very well yet.

> Use client-side (***not*** server-side) image maps. Alt tags don't work with server-side image maps.

That's it. You'll probably learn how to do a lot more as you go along, but even if you only do what I've covered here, you'll be doing a pretty good job.

Hopefully in five years I'll be able to just remove this chapter and use the space for something else because the developer tools, browsers, screen readers, and guidelines will all have matured and will be integrated to the point where people can build accessible sites without thinking about it.