

website owner's manual

the secret to a successful website

Sample Chapter

paul boag

Foreword by Ryan Carson



manning



Website Owner's Manual

by Paul A. Boag

Chapter 7

© 2010 Manning Publications

CONTENTS

	foreword	xv
	preface	xvii
	about this book	xix
	author online	xxii
	about the author	xxiii
1	The secret to a successful website	1
2	Stress-free planning	17
3	The perfect team	40
4	Differences over design	63
5	Creating killer content	88
6	User-centric design	106
7	Ensuring access for all	126
8	Taking control	149
9	Decoding technobabble	171
10	Driving traffic	194
11	Engaging your visitors	220
12	Planning for the future	241
	Index	261

7

Ensuring access for all

In this chapter

➤ **Identify the cowboys**

Learning from history

Understanding the consequences of poor code

- *Coding for multiple browsers*
- *Bloated code*
- *Hard-to-maintain code*
- *A culture of redesign*

Learning a better way to build websites

➤ **A matter of style**

Improving printing

Capturing the emerging market

Responding quickly to change

➤ **Never turn away users (or Google)**

Handling the expense of accessibility

Achieving increased traffic with minimal effort

➤ **Exceed your legal obligations**

➤ **Create an accessibility policy**

Establishing your long-term accessibility goal

Having a roadmap for overcoming common problems

- *Poorly described images*
- *Badly labeled links*
- *No alternatives to media*
- *Reliance on JavaScript*
- *Hard-to-read text*

Testing accessibility

Maintaining accessibility

Dealing with complaints

Being a geek, I love *Star Trek*. I'm sorry, I can't help it. I enjoy the amount of thought that goes into it. Take the difference between the original *Star Trek* series and *Star Trek: The Next Generation*. The original series was like the Wild West. Kirk was a futuristic gunslinger exploring the galaxy, pushing back frontiers, and breaking all the rules. In *Next Generation*, the federation had grown up, and those pioneering days were over. It was about politics and alliances. New discoveries were happening, but at a slower pace.

I bring up *Star Trek* in order to make a comparison. The early days of the web were like the original series: there were no rules, everything was new, and we were flying by the seat of our pants. Now, the web has matured. Like *Next Generation*, you have your own prime directives. These include accessibility, standards, and legal requirements. Web design has grown up a lot. Unfortunately, a lot of Kirks remain, building websites like it's 1999.

This chapter will help you make sure your website is accessible to everyone from search engines and disabled users to those wishing to print your web pages or access them from a mobile devices. It will explain best practice and ensure that you don't get caught by cowboy designers peddling out-of-date techniques.

IDENTIFY THE COWBOYS

In order to identify cowboy designers and create an accessible site, you need to understand best practice. Before I can help you do that, I need to begin with a history lesson.

Learning from history

When Tim Berners-Lee proposed the World Wide Web back in the early 1990s, he couldn't have imagined what we have today. Early web pages were purely textual. There were no images and certainly no video. We also had no substantial control over layout or colors. A web page was expected to describe the *meaning* of content, not to dictate its appearance. The appearance of the document was controlled by the browser.

For example, a heading was marked up with a heading tag like so:

```
<h1>Identifying the cowboys</hi>
```

And a paragraph was marked up using a P tag:

```
<p>While a paragraph would be marked up using a P tag...</p>
```

The browser interpreted this markup by displaying headings in a larger font and putting carriage returns between paragraphs. In short, the web page described the content's *semantic* meaning.

As the web grew in popularity, it became more than a repository of information. It began to be perceived as a marketing tool, and website owners wanted control over the appearance of their pages. Browser manufacturers obliged by introducing support for adding images, controlling colors, and setting fonts. The markup now described the appearance of the page as well as the content. Design and content began to mix.

Even this wasn't enough to satisfy the marketers. They wanted to replicate print designs on the web. This led to the abuse of markup. Instead of tags being used to

What's New, June 1993

June 27, 1993

Digital Equipment Corporation is running a [Web server](#) from their [anonymous FTP server](#). Included are pointers to [DEC product information](#), [Ultras and OSF/1 FAQs](#), [DEC research reports](#), and more.

The Army Research Laboratory is now running a [Web server](#). Included is information on the [ARL Scientific Visualization effort](#), a picture of the [arrival of a KSR-1 supercomputer at ARL](#), and more.

Information on the [Front Range Consortium](#) is now online -- here's their [Web server](#). Members of the Front Range Consortium include [CAPE](#), [McABSC/D](#), and [WAAA/PSI](#).

The [Navy Research Laboratory Advanced Concepts Group](#) is now online.

June 25, 1993

A [Web server](#) has been installed at the Centre Universitaire d'Informatique of the University of Geneva. Information about various research groups at the CUI is available, as well as a number of other experimental services.

June 24, 1993

The Institute for Theoretical Physics at State University of New York at Stony Brook is now running a [Web server](#). Included is [online Institute news bulletins](#), a directory of [people](#), and [local system documentation](#).

HyTelnet 6.5 is now online; see [here](#).

If you haven't tried it yet, take a look at the [Web server](#) running inside [JazzHouseMOO](#). See particularly the [subject browser](#). (For those unfamiliar with the term, a MOO is a "consensual text-based virtual reality", aka MUD or Multi-User Dungeon, wherein people interact with one other in a computer-enabled world. Since these systems commonly feature rich and extensible programming environments, it is possible to build Web, Gopher, and other servers (and clients) directly into the online virtual world. Another example of this is the [Gopher server](#) running inside the [Axiomair MUD](#); more examples are [here](#).)

June 23, 1993

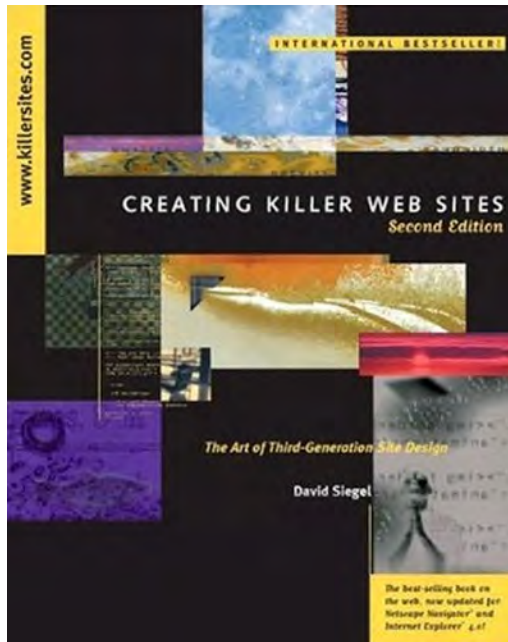
A new server at Georgia Tech is [here](#); see in particular [resources by subject](#). Most of the stuff in there is off limits to off-site people -- banner. But not a hypertext version of [The Whistle and Information on their Graphics, Visualization, and Usability Lab](#).

June 22, 1993

A new and greatly improved [HTML printer](#) is now online, courtesy of the Publications group at NCSA. Comments to [pubs@nca.nimc.edu](#).

Carnegie Mellon has announced their Web server; here's the ["Front Door"](#); here's the [home page](#). ("Front door" -- interesting metaphor, that.) Interesting things on their server include a hypertext [CMU technical reports archive](#), an index of online [reference works](#), an index of [online journals](#), some [personal home pages](#), and more. Also, they're keeping us Mosaic developers in line with their [own internal list of Mosaic bugs](#). (Bugs? *Er?*)

Early markup provided no control over design.
It existed to describe content, not appearance.
The browser decided how to display the page.



In 1997, David Siegel's book *Creating Killer Web Sites* (New Rider) popularized the use of tables for controlling web-page layout. At the time, this appeared to be positive; but in hindsight, it proved detrimental.

describe the meaning of content, they were selected because the browsers displayed them in a particular way. For example, web designers chose an H1 tag not because they wanted to mark up a heading but because they wanted to make text large and bold.

The biggest abuse was in the use of the table tag. The table tag was originally created to display tabular data, similar to most spreadsheets. Web designers quickly discovered that tables could be used to position content on a page with almost to-the-pixel control.

Admittedly, this approach worked. Website owners got the print-like designs they wanted. Software automated the process of creating this spaghetti code and kept development costs low. Where, then, was the problem?

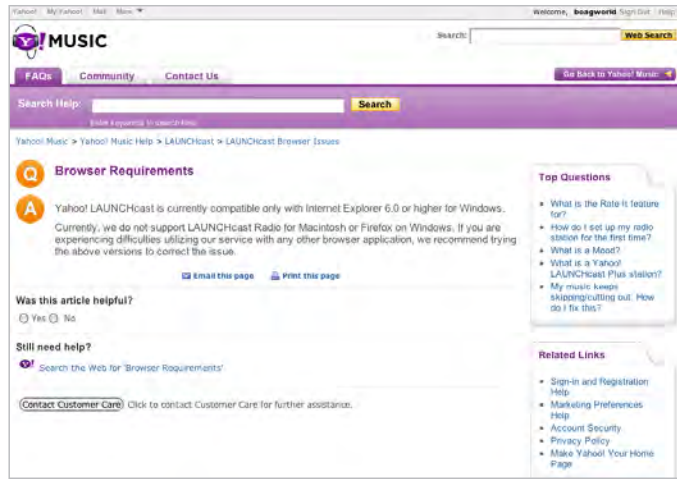
Understanding the consequences of poor code

Unfortunately, design control came at a cost. The complicated, messy markup created by table-based design had ramifications. Web pages became

- Browser specific
- Bloated
- Hard to maintain

Coding for multiple browsers

With browsers displaying this “bastardized code” in slightly different ways and offering support for their own proprietary tags, it became increasingly hard to build sites that were accessible by all. Designers were often forced to present different versions of a site to each browser, thus increasing development time. Designers working with limited budgets sometimes gave up and supported only a single browser. If the user happened to be using a competing browser or an old version, they were effectively excluded from the site. These badly coded websites sometimes excluded users accessing the internet via an alternative device such as a screen reader (used by people with visual impairments). Table-based code could also exclude those with a slower connection.



Yahoo! Music effectively excludes users who aren't using the latest browser.

Bloated code

Because HTML markup was never intended to produce complex designs, large amounts of additional code were required. The average web page swelled in size and took a considerable time to download.

When broadband arrived, web designers believed that download speed was no longer an issue. But the rise of broadband was accompanied by an increase in web access via mobile devices. These devices typically only have dial-up connection speeds.

Download size is also still an issue for larger, more heavily trafficked websites. A cost is associated with data served from a website. When downloads exceed a certain limit, the hosting provider may start to charge. Tiny amounts of data can make a real difference when they're downloaded many thousand times.

The cost of poor code isn't just financial. There is also a cost in work hours.

Hard-to-maintain code

Complex code was hard to maintain. Even the simplest change, such as altering the size of text, had to be made many hundreds of times across every page on a website.

Website owners once paid web designers ridiculous fees to make changes to content

when they should have been able to do it themselves. Many sites have unnecessary content-management systems (CMSs) because the code was too complicated to understand. Sites became obsolete as new browsers were released and broke existing code. It was often more cost effective to throw out a site and start again than to fix what was there.

A culture of redesign

The difficulties in maintaining these websites led to a culture of redesign. Every few years, an organization became frustrated with its website and got a web design company to fix the site. Unfortunately, this new site was often just as badly built and also degraded over time as new browsers were released and content wasn't kept up to date. The process then repeated itself.

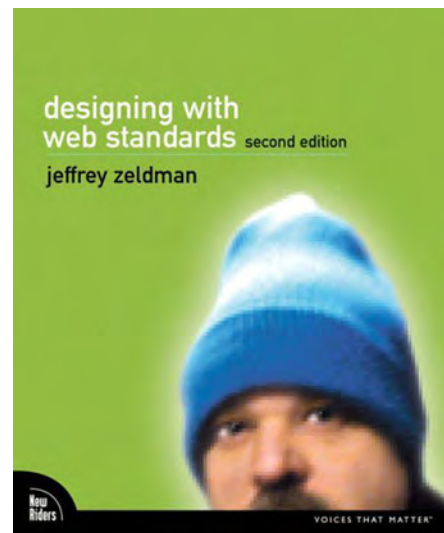
So far, I've spoken about these problems in the past tense, implying that websites are no longer built this way. But many still are. For a variety of reasons, web designers choose not to keep up with best practice. It falls to you as the client to set the standard of work you require. To do this, you need to understand standards-based design.

Learning a better way to build websites

While web designers were bastardizing markup to suit their needs, Tim Berners-Lee and a group of industry experts (the World Wide Web Consortium [W3C]) were working on a solution to the problem. They proposed a complete separation of content from design. The markup would return to its original role of defining the meaning of content. Meanwhile, a separate file would describe to the browser how that content should look. This file was called a cascading



Fixes were required as browsers updated. Unfortunately because of the complexity of the code these fixes were expensive.



Designing with Web Standards by Jeffrey Zeldman (Peachpit, 2006) is the definitive introduction to standards-based design. It's ideal for explaining the benefits of standards to designers and developers.

stylesheet (CSS). This separation of content from design provided the design control that website owners demanded, while avoiding the pitfalls of other techniques.

At first, browser support was limited. But thanks to the campaigning of groups like the Web Standards Project, that situation quickly changed. A new generation of web designers emerged, dedicated to building websites that conform to these new best practices. These websites are more likely to be accessible, faster to download, and easier to maintain. They provide a host of additional benefits that are explored in the next section.

Unfortunately, not all web designers are so progressive. Many still produce websites that mix content and design and ignore accessibility. With many clients still ignorant of best practice, these developers have little motivation to change their ways. These clients only care that their site looks OK on their computer—but you should consider the broader picture. Toward that end, I'll endeavor to clearly explain the benefits of standards-based design. Let's begin with control over styling.

A MATTER OF STYLE

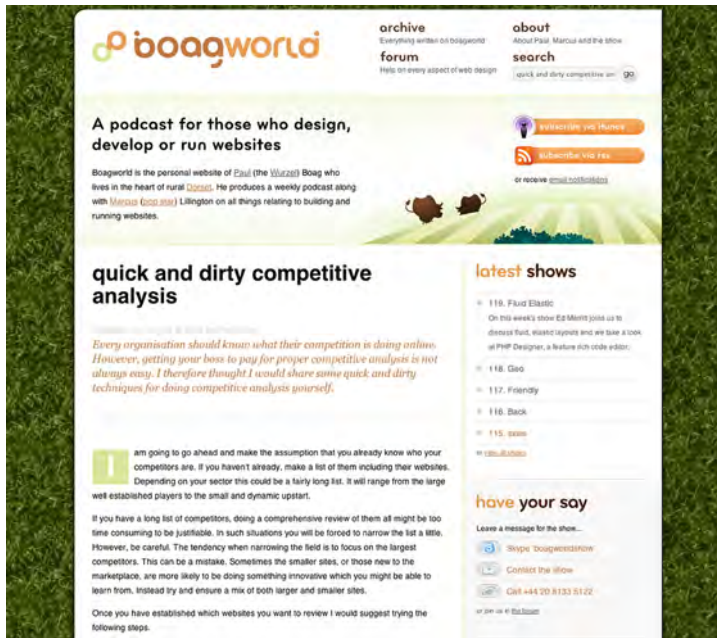
The greatest benefit of this new approach to building websites is born out of the separation of design from content. With all design being defined from a single file (the CSS), it's simple to change that file and give your site a different look. The content remains the same, but the site's design can change radically. This offers a host of possibilities, not least when you're printing.

Improving printing

If you've ever printed a web page, you know that printing on the web is less than satisfactory. You have to deal with two fundamental flaws:

- *It wastes paper and ink by printing screen elements that the user doesn't need.* Why print the navigation or interface graphics? These are needed while you're interacting with a website, not for reading a piece of paper.
- *Many websites fail to print properly.* This can manifest in many ways, but the most common is content being truncated down the right side of the page.

Standards-based design resolves these problems by allowing you to specify a different look and feel when printing. By swapping the stylesheet, you can change to a design that prints perfectly and removes the unwanted screen elements. For example, take my



This is how the Boagworld website (<http://www.boagworld.com>) looks when viewed through a web browser.



This is how the same site looks when printed.

The screenshot shows the Washington Post website with several annotations in yellow callout boxes:

- Top Left:** "Printing search and navigation is unnecessary because they can't be used in printed format." (points to the search bar and navigation menu).
- Top Center:** "Printing banner advertising takes up valuable ink and angers users." (points to the Verizon banner).
- Top Right:** "Content on the right side of the page isn't printed." (points to the TOC, In F, GOI, U.S., and Advert links).
- Bottom Left:** "Secondary navigation is redundant in printed form" (points to the 'Immigration Issue' sidebar).

The page content includes the main headline "Obama to Deliver Patriotism Speech" by Jonathan Weisman, a "THE FACT CHECKER" section, and a sidebar with "Immigration Issue" and "Obama Will Testify". The URL at the bottom is <http://blog.washingtonpost.com/the-trail/2008/06/30/obama-to-deliver-patriotism-sp.html> and the date is 6/30/2008.

Many websites print poorly. They cut off content and waste both paper and ink. Although users can correct these problems with plug-ins and settings, you should always ensure that your site prints correctly.

own site at boagworld.com. When people print one of my blog posts, all the secondary content and navigation are removed. The site prints only the information users want.

Separation of content from design also helps when you're catering to the emerging mobile sector.

Capturing the emerging market

Cameron Moll, in his e-book *Mobile Web Design*, states that by 2010 it's anticipated that there will be 4 billion mobile-phone subscribers worldwide. That is an astounding 59% of the entire planet's population. The cell phone has existed only 35 years, and yet the cell-phone industry has sold 2.7 billion units. Compared with 850 million personal computers sold over 30 years, this is amazing growth. In the U.S. and U.K., access to the web from a mobile device accounts for between 17% and 19% of web usage. These figures are set to increase.

In chapter 12, "Planning for the future," we'll look at harnessing this emerging market. For now, all you need to know is that separating content from design is a key part of the process. Separation lets you deliver a mobile-friendly stylesheet to mobile devices without the need to maintain multiple sites.

The ability to swap styles is only the tip of the iceberg. Separating content from design also allows your website to adapt based on circumstances.

Evernote on the web



Evernote on an iPhone



Evernote on a cell phone



Evernote.com uses standards to help bring different visual experiences to a variety of devices.

Responding quickly to change

Traditionally, websites have been difficult to change. When content and design are mixed, a small change like altering the default font size requires every page on the site to be edited. When you use standards, this is no longer the case—and that offers a wealth of advantages.

Imagine you're launching a new advertising campaign that has a different look and feel from the rest of your brand. Using standards-based design, you can change the entire site to match this design and swap back when the campaign is over.

You can also style the site depending on where the user is referred from. If they come from a partner's website, your site can be rebranded to match their style. This provides a seamless experience for the user.



The real power of standards isn't in complete redesigns but in the ongoing evolution of your site. Standards make it easy to continually improve your site based on user feedback. If people complain that links are hard to read, then you can quickly change them. If users can't find the search bar, you can restyle it. The ability to make quick, incremental changes in response to user requirements provides a competitive edge.

Standards also provide tangible improvements to the accessibility of your site.



Jason Santa Maria uses stylesheet switching to customize the design of his site to suit the content of each page.

NEVER TURN AWAY USERS (OR GOOGLE)

A common response I hear when talking about web accessibility is that it “doesn’t apply to us because we don’t have any blind users.” Setting aside the obvious absurdity of this statement (you won’t have blind users if your site is inaccessible), it demonstrates a fundamental misunderstanding of what web accessibility is about.

Web accessibility shouldn’t stop at access for the blind. In fact, you shouldn’t focus solely on access for the disabled. Everyone should have access to the web whether they use a screen reader because of a visual impairment or a mobile device with poor connection speeds and text-only support. Access should be available regardless of device, connection, or disability.

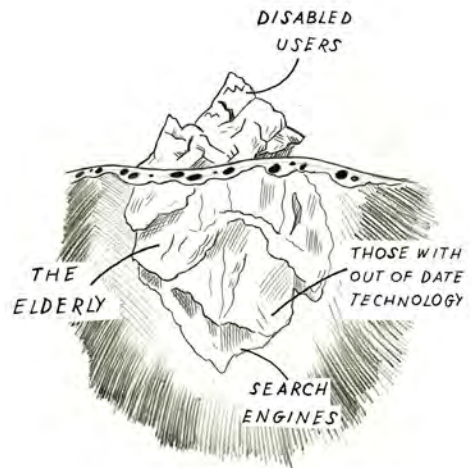
When you have this mindset, the importance of separating content from design becomes more obvious. Although your content remains the same, its presentation needs to change based on user requirements. A mobile device with a small screen requires a different design solution than a desktop computer. A screen reader requires no design at all, whereas someone with low vision may want larger text. All of this is straightforward when design and content are separate.

Recognizing that accessibility is about access for all brings even greater realizations—particularly from a financial perspective.

Handling the expense of accessibility

A big objection to providing an accessible site is the expense. Quotes to make a site “accessible” often seem disproportionately high when compared to the financial returns from a minority disabled audience.

In reality, this reasoning is flawed. First, the high cost of “making a site accessible” is normally associated only with sites built using out-of-date techniques. When a site separates content and design, it’s inherently more accessible. Also, the cost of further improving that accessibility is significantly less. Second, as we’ve already established, accessibility isn’t just about the disabled; return on investment will be significantly higher than if you focus only on disabled users.



Disability is the tip of the access iceberg. Accessibility also including meeting the needs of the elderly, those with out-of-date technology, and the unique requirements search engines have when indexing your site.

In the United Kingdom alone, it's estimated that those registered as disabled have a spending power of more than \$160bn. Add to this those not registered as disabled but with physical or cognitive conditions that affect their use of the web. The elderly are a great example of this audience: as you age, your vision declines, as do your motor skills, making websites increasingly difficult to use.

Even if you stop there, this constitutes a significant audience that you're potentially turning away from your website. When you add those using dial-up connections, working on older computers, or lacking the latest plug-in, it quickly becomes apparent that access for all can't be ignored.

Imagine turning away customers from a restaurant because they're too old or suffer from color blindness. You wouldn't do it. Why then do you turn people away on the web?

You aren't just turning away users. If your site isn't built using best practices, you may also be turning away search engines.



A restaurant owner would never turn away paying customers, so why should your website?

Achieving increased traffic with minimum effort

In many ways, a search engine like Google is the ultimate disabled user. It can't see, and has only limited support for more advanced web features such as video and audio. It's easy to build a site that is either totally or partially inaccessible to Google.

Search engines are interested in only one thing: providing relevant results to their users. They only look at the content. They don't care about design or advanced web features.

If you want your website to rank highly on search engines, you need to ensure that they can access your content. Building with best practices and accessibility in mind will do that.

By separating design from content, you make it easier for a search engine to catalogue your site. If you mark up that content semantically (describing the headings, and so on), you make it easier for search engines to understand what the page is about.

Finally, many of the techniques used to improve disabled access also help search engines. For example, blind users can't see images, so a hidden description is associated with each picture. This is called an **ALT** attribute. It tells the user what is contained within the image—and it tells search engines the same thing.



```

```

The ALT attribute describes the content of an image. This is important for visually impaired users but also helps with search-engine indexing.

Many organizations pay large sums of money to improve their ranking on search engines while ignoring accessibility as “not cost effective.” If they spent the money on best practices, they could have both. In chapter 10, “Driving traffic,” we’ll explore more ways to promote your site. For now, you need to be aware that accessibility can help with your search-engine rankings.

Accessibility can provide financial benefits by increasing the amount of traffic going to your site and allowing more of those visitors to gain access. But that isn’t the only reason to worry about accessibility. There are also potential legal obligations.

EXCEED YOUR LEGAL OBLIGATIONS

Fear of litigation is the most common motivating factor for addressing a website’s accessibility. From lobby groups campaigning for disabled rights to web designers trying to drum up business, there is no shortage of people saying your site is breaking the law. Most of these claims are extreme; and although some could potentially be true, they do little to encourage best practice.

I’m not a lawyer and have no intention of giving legal advice about your liability. The varying legislation on web accessibility worldwide makes that impossible. What I can share with you is my experience and what I have observed online.

You probably have a legal obligation to provide an accessible website. Whether it's Section 508 in the U.S. or the Disability Discrimination Act in the U.K., most countries have some form of legislation to address the issue of online accessibility. But even if your website contravenes the legislation, you won't necessarily be taken to court.

A more likely scenario is that you'll receive a complaint about some aspect of your site. How you respond to that complaint will dictate whether you end up in court. Failure to respond quickly or take the complaint seriously could lead to litigation. If you respond quickly and apologetically, then chances are the user will go away happy.

Am I proposing that you ignore web accessibility until somebody complains? Certainly not. I hope I've already demonstrated good reasons to address accessibility beyond legal requirements. I merely wish to dispel the fear-mongering that surrounds this subject.

This kind of fear-mongering demands all or nothing. You must comply with legislation today or face the consequences. This can seem overwhelming, and many people give up without trying. They choose instead to take the risk that nobody will draw attention to the failings of their site.

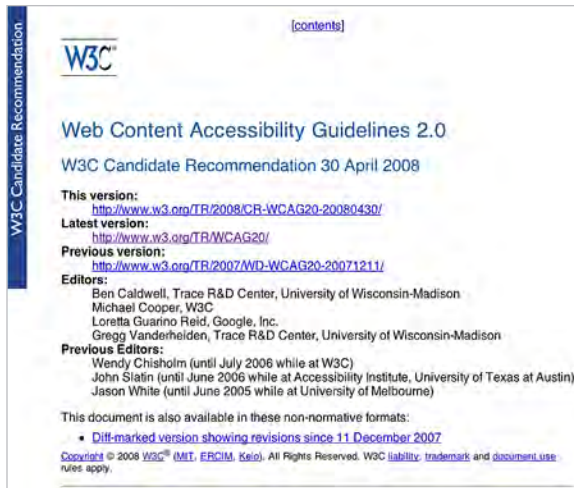
Instead, I believe website owners should start small and improve accessibility over time. A large proportion of accessibility problems can be overcome with a few simple fixes.

There is no lack of people suggesting ways to ensure your site's accessibility, from government legislation to pressure groups and disability charities. The best solution lies with the World Wide Web Consortium (W3C). In addition to producing the specifications for HTML, CSS, and other aspects of standards-based design, the W3C has also produced extensive guidelines on accessibility. That is why accessibility and standards are so closely linked.

The guidelines produced by the W3C have become the template for almost all other accessibility advice. The first guidelines published by the W3C came out in 1999 and were referred to as WCAG 1. Since then, the web has moved on considerably, so the W3C has produced a second version called WCAG 2.



How you handle complaints will dictate whether your organization ends up in court.



The Web Content Accessibility Guidelines 2.0 (WCAG 2) are the definitive template for web accessibility going forward.

At first glance, the W3C guidelines can be intimidating. The documentation associated with them is extensive and highly technical in places. But the guidelines themselves are relatively easy to understand. They’re broken down into four principles regarding content:

- *Perceivable*—Elements on your website must be presentable to users in ways they can perceive.
- *Operable*—User must be able to navigate and use your website.
- *Understandable*—Your website must be easy to understand.
- *Robust*—Content must be robust enough that it can be accessed by a variety of different devices.

These are all common sense. Then, each of the four areas is segmented into specific ways you can achieve it. For example, under “Perceivable” is a guideline that reads “Provide text alternatives for any non-text content so that it can be changed into other forms people need, such as large print, Braille, speech, symbols or simpler language.”



W3C guidelines appear intimidating, but the core principles are easy to grasp.

Each guideline is written in language that is relatively easy to understand, and I recommend you take the time to read them all. The guidelines are divided into individual success criteria, but this technical detail isn't relevant to you as a website owner.

Understanding the guidelines is important for two reasons. First, it makes it easier to ensure that any web designer you hire knows the latest accessibility techniques. Shortcomings are obvious when you're aware of what is required. Second, you need to understand the basics of accessibility if you wish to create an accessibility policy for your organization.

CREATE AN ACCESSIBILITY POLICY

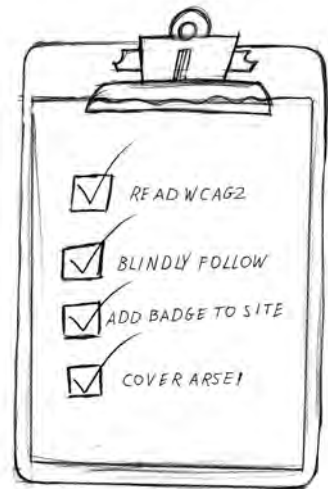
As I've said, many organizations embrace accessibility not due to a desire to improve access, but from a fear of litigation. This leads to a "butt-covering" mentality. They fixate on a set of guidelines (like WCAG) and blindly check every box until they have fully conformed with the specification.

This approach is flawed because it's organizationally focused rather than user focused. Following generic guidelines that may or may not apply to your users' specific needs is wasteful and serves nobody.

A better approach is to use WCAG as a starting point for the creation of an accessibility policy. This outlines your organization's approach to dealing with issues of accessibility and should include the following:

- Your ultimate objectives in terms of W3C guidelines
- A roadmap for reaching these objectives
- A process for testing compliance with these objectives
- A plan for maintaining the accessibility of your site over time
- A procedure for responding to complaints

This policy shouldn't be drawn up in isolation but in consultation with your developers



Many organizations only choose to comply with the W3C accessibility guidelines in order to avoid prosecution.

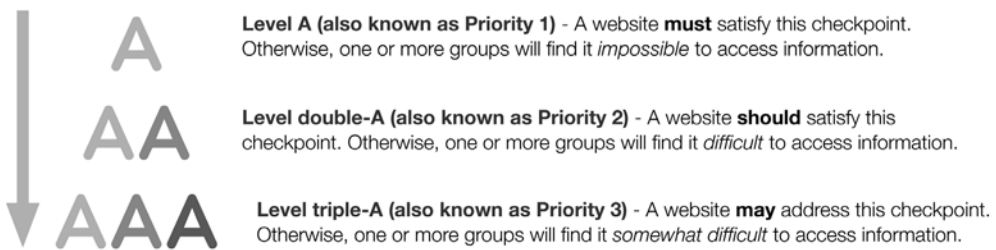
(who must implement the objectives) and your content providers (whose help will be vital if your site is to remain accessible). Let's look at each of the elements that appear in your accessibility policy.

Establishing your long-term accessibility goal

Every accessibility policy should have an end goal in mind. This objective will probably change over time, but there is value in documenting your current aim. What that aim should be comes from discussion with developers, content providers, and end users. But it will probably be based on some aspect of WCAG 2.

Each WCAG 2 guideline is broken down into one or more success criteria. These criteria are rated according to their level of conformance. There are three levels: A, AA, and AAA.

The three levels of web accessibility



Traditionally, organizations have decided to reach a certain level across all guidelines. For example, a company may aim to make an entire site Level AA compliant. Depending on your circumstances, this may prove difficult to achieve. A better approach is to aim for a minimum of Level A across the board but seek to comply to higher levels of accessibility on some guidelines. This more tailored approach takes into account the varying requirements of your business, content, and audience.

Of course, identifying an end goal is one thing. Getting there is another.

Having a roadmap for overcoming common problems

Even achieving the most basic level of accessibility (Level A) can be challenging if you're implementing it on an existing website. It is far cheaper and easier to plan with accessibility in mind from the outset. But when you don't have that luxury, you don't need to immediately implement Level A. Your accessibility policy can outline a roadmap for achieving longer-term goals.

The *Pareto principle* or *80/20 rule* (http://en.wikipedia.org/wiki/Pareto_principle) states that for many events, 80% of the effects come from 20% of the causes. This holds true for accessibility, where a small number of issues cause the vast majority of problems. It's logical to start any roadmap by resolving these issues first—but what are they? That is a subjective question, but here are the most common problems I encounter:

Poorly described images

I've mentioned that images should have associated **ALT** attributes. This benefits both visually impaired users and search-engine placement. But the problem of poorly described images isn't due only to a lack of description. It's also caused by badly written descriptions.

Because many people realize the benefit of **ALT** attributes for search-engine placement, they fill these descriptions with keywords and make them overly long. You should make sure all content images have an **ALT** attribute and that it clearly describes what is being shown in a single sentence. Longer descriptions can be annoying when read back by a screen reader.

But it isn't just images that are labeled badly; so are links.

Badly labeled links

The text in a link should describe that link without context. Screen readers can read all the links on a page as a single list, which helps users quickly navigate without listening to the entire page. The problem arises when a link is entitled “click here,” which doesn't give the user any information about where the link leads. A better link would read “latest news.” Where a longer description is required, a **title** attribute (similar to an **ALT** attribute) can provide more information.

Using descriptive links helps not only screen-reader users but also users who are quickly scanning a page looking for the next link to follow. And search engines use the content of a link as a way of judging what the page linked to is about.

In addition to describing links and images, you also need to consider other forms of media.



If a link is labeled “click here,” it's meaningless out of context. For visually impaired users, this can be frustrating.

No alternatives to media

When you're using video, audio, or any form of media that requires additional plug-ins, you need to provide an alternative version. This alternative should be in the form of either a transcript (in the case of audio) or captions (in the case of video or other media where visuals and audio are synced).

At first glance, this seems like a massive undertaking. But a number of services like Casting Words provide transcription at a reasonable rate, and tools like Overstream (<http://www.overstream.net/>) can help you create captions.

It's important to provide these alternatives because users may not be able to perceive the content due to technological limitations or disability. This is also true for JavaScript.

Reliance on JavaScript

JavaScript is a programming language that is used to achieve many of the interactions you see on websites. From pop-up windows to services like Google Maps, JavaScript is amazingly flexible and heavily used.

JavaScript itself isn't inaccessible. It exists to add interaction and behavior to a website in the same way HTML provides content and CSS provides design. The problem is in the implementation.

Not everyone has access to JavaScript, and search engines regularly ignore it. It's therefore important that all content is accessible even when JavaScript isn't available. The most common problem is using JavaScript to create navigation and other links. If JavaScript isn't available, it's impossible for users to follow those links to the content beneath. Similarly, when JavaScript is used to add content to a page, this content becomes inaccessible if JavaScript is disabled. Never rely solely on JavaScript as a method of accessing content.

The final accessibility mistake I see regularly is preventing users from resizing text.

CastingWords
TRANSCRIPTION SERVICES

Transcription Store
Here is what you get with every transcription order from CastingWords:
All transcription is done by people, not machines.
Transcriptions are delivered in plain text, HTML, and RTF formats.
You get an RSS feed of all of your transcriptions.

Budget Transcription:
90.75 minutes
Our lowest transcription service. While not having a guaranteed turnaround time, transcription often takes a month or more. If you have a deadline we highly recommend that you use our 6 day or 1 day transcription services.

6 Day Transcription:
61.50 minutes
Completed in 6 days or your money back. Transcriptions is done by highly rated transcribers.
Experience Service page, with more details.
Terms of Service (MPL): The service is only for high quality audio.

1 Day Expedited Transcription:
52.50 minutes
Completed in 24 hours or your money back. Transcription is done by our most highly rated transcribers.
Experience Service page, with more details.
Terms of Service (MPL): The service is only for high quality audio.

Submit a URL:

Or upload a file: no file selected

This audio is difficult (60.75/m) is your audio difficult?
 Not Transcription (60.75/m)

*File size limits to 2 MB, MP3 or 320 kbps. The actual price for a MP3 or WAV file. You pay for the length of the file - upload 100 seconds and you submit. We also offer a professional transcription service, which we transcribe every new product by you and it to your need.

New Features
24 June 2008: Our new [order tracking page](#) makes it easier to follow your order.
16 May 2008: Account holders can now use [FTP to upload files](#).

The New York Times [article](#) and The Economist [article](#) (article subscription required) have both mentioned our company in articles.

[Store](#) | [Order Tracking](#) | [View Cart](#) | [Privacy Policy](#) | [Jobs](#) | [Professional Services](#) | [FAQ](#)

If you have any questions or concerns please email support@castingwords.com

Casting Words (<http://Castingwords.com>) converts audio to text. This is just one of the services that can help your site become more accessible.

Hard-to-read text

By default, all major browsers let users control text size. This is required for users with less than perfect vision. Most visual impairments require font sizes to be increased. But some people need smaller text to fit better within a limited field of view.

Also, many visual impairments are made worse by poor color contrast between the text and background colors.

Although browsers provide the ability to resize text, and it's relatively trivial to design with sufficient contrast, many web designers fail to build with this in mind. When the user resizes text, the design breaks and becomes unreadable. Poor color combinations increase the problem. There is no good reason for this beyond laziness. Ensure that your designers consider these issues when developing your site.

By addressing these five problems, you can dramatically improve the accessibility of your website. None of these issues is particularly hard to overcome, and the financial investment is minimal. You'll increase the traffic to your site and the number of visitors able to successfully navigate it.

But an accessibility policy shouldn't just address quick fixes. It should also provide a comprehensive approach to improving and maintaining site accessibility. To do this, it needs to include a degree of accessibility testing.

Testing accessibility

How you intend to test the accessibility of your site should be a fundamental component of any accessibility policy.

Most organizations rely too heavily on automated services. These online services claim to test web accessibility but can only carry out a basic review. For example, they can't ascertain whether descriptions in ALT attributes on images are meaningful or merely stuffed with keywords to improve search-engine placement. They're unable to test some guidelines at all—and this is particularly true with WCAG 2.



The BestBuy site (<http://www.bestbuy.com/>) doesn't adapt well when users enlarge the text. Screen elements are forced out of position, and text overlaps.

W3C Home

Web Accessibility Initiative (WAI) Home

Introducing Accessibility

Guidelines & Techniques

Managing Accessibility

Evaluating Accessibility

- Preliminary Review
- Conformance Evaluation
- Specific Contexts
- Involving Users
- Selecting Tools
- Tools Search
 - Complete List
 - Simple Search
 - Advanced Search
 - Combined Expertise
 - Reporting Template

WAI Groups

About WAI

WAI Site Map

Help with WAI Site

W3C/ARIA Transitions

W3C/ARIA About RSS

W3C Search

Web Accessibility Evaluation Tools: Overview

Overview Complete List Simple Search Advanced Search

Getting Started

Web accessibility evaluation tools are software programs or online services that help determine if a Web site meets accessibility guidelines. While Web accessibility evaluation tools can significantly reduce the time and effort to evaluate Web sites, **no tool can automatically determine the accessibility of Web sites.**

Before using this list of evaluation tools, please read "[Selecting Web Accessibility Evaluation Tools](#)", which provides background for using the search effectively.

Finding Evaluation Tools

The W3C WAI list of Web accessibility evaluation tools is available through:

- [Complete List](#) - to show all tools in the list maintained by WAI
- [Simple Search](#) - to find tools using common search criteria
- [Advanced Search](#) - to find tools using detailed search criteria

Disclaimer

W3C does not endorse specific vendor products. Inclusion of products in this reference list does not indicate endorsement by W3C. Products and search criteria are listed with no quality rating. Information in this database (including the search criteria) reflects claims made by tool developers, vendors, or users; and it can change at any time. W3C does not verify the accuracy of these claims. The list is not a review of evaluation tools, nor a complete or definitive listing of all tools.

Page Contents

- [Getting Started](#)
- [Finding Evaluation Tools](#)
- [Disclaimer](#)
- [About the Tools List](#)

Although they're sometimes misused, accessibility checkers have a role to play. For a list of tools, see <http://www.w3.org/WAI/ER/tools/>.

There is a place for automated testing. It's useful for identifying potential problems across a large site, but you should never use it in isolation.

User testing with disabled users is far more effective than any amount of automated testing. Granted, finding appropriate disabled users isn't easy, especially when you're faced with a wide range of potential disabilities. Fortunately, a number of organizations can help you with recruitment. They can arrange testing for you and even advise you on how to solve any problems that arise.

Paying for real users may be beyond your budget. If this is the case, ensure that somebody is made responsible for regularly testing the accessibility of your site through a mixture of manual quality control and automated testing.

Whatever approach you adopt, make sure it's documented in your accessibility policy, including how often you intend to test. Frequent testing is important because although it's easy to launch an accessible website, such a site can be difficult to maintain.

Maintaining accessibility

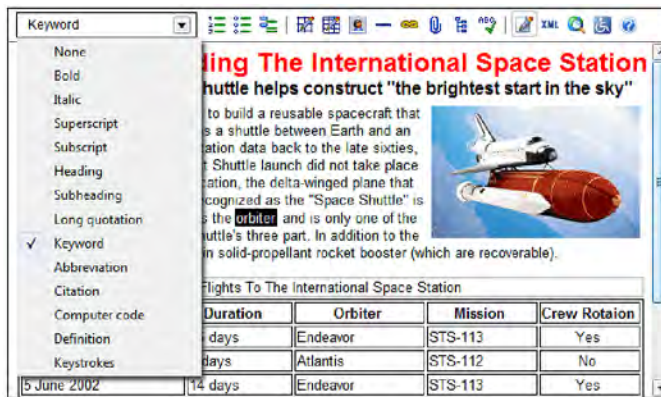
Maintaining the accessibility of your site can be problematic if it's being updated on a regular basis. It's even more complex if multiple people are involved in adding content. Your accessibility policy should address how to maintain accessibility over the long term. You can do so in three ways:

- Regular testing (as explained earlier)
- Training
- Using the right technology

The most important of these is training. Ensure that your development team and content providers understand the basic rules of accessibility. Content providers may only need a set of simple rules to follow. Your development team, on the other hand, will need a deeper understanding and may require formal training.

Unfortunately, a good understanding of accessibility doesn't help when technology fails to produce accessible code. This is especially important when you use a CMS. Make sure your CMS can output clean, accessible code and that wherever possible it enforces accessibility on content editors (such as requiring ALT attributes to be defined for images). This is certainly an important point to consider when procuring a CMS.

No matter how good your testing and maintenance plan, problems will arise, and you may receive complaints. How you respond to those complaints will determine whether you find yourself in court.



One accessibility weakness of many CMSs is the editor. Although the CMS produces accessible code, the editor allows content providers to undo this good work. Consider using an editor like XStandard (<http://xstandard.com>), which was built with accessibility in mind.

Dealing with complaints

Your accessibility policy should establish the following:

- *Who is responsible for dealing with complaints*—If this isn't clearly defined, complaints often remain unanswered. If a user doesn't receive a response because nobody saw it as "their job," they're more likely to turn to litigation.
- *How quickly your organization should respond to initial complaints*—The person responsible for responding to complaints often has other responsibilities. An email from an angry user may not come high on their list of priorities. To ensure a quick response, set targets in the accessibility policy. This will prevent emails from being forgotten.

- *The process for addressing a complaint*—What happens when a complaint is received? Who estimates the amount of work required to fix the problem? Who signs off on the expenditure? Who is contacted if legal advice is required? These kinds of questions should be answered in your policy.
- *What to do if the complaint can't be addressed*—Sometimes an accessibility issue can't be fixed, or the cost of doing so would be prohibitive. In such cases, a decision must be made about how to respond. This is where it's necessary to open a dialogue with the person complaining. Ask them how they believe the problem could be resolved. Offer them alternative methods of getting the same information (printed brochure, phone, and so on). If you can explain the problem and offer a compromise, most people will be happy. The important thing is to address the complaint and not just hope it goes away.

Dealing with complaints quickly and efficiently should not only avoid litigation but also create improved customer loyalty. Users who encounter problems that are then resolved quickly view a website in a more positive light than if they hadn't encountered the problem at all.

Next actions

This chapter started by demonstrating the importance of accessibility and standards. It then went on to identify elements you should address in these areas. The emphasis has been on informing, rather than encouraging specific actions, because the implementation of standards and accessibility is done by your development team rather than yourself. That said, I encourage you to complete the following three steps:

ACTION 1: *Get your team on board.* The first and most important step is to ensure that your developers and content providers understand the importance of standards and accessibility. If you're recruiting an outside agency, be sure they're using modern techniques. If you have in-house staff, use the content of this chapter to argue for best practice and then consider training to bring people up to speed. If you don't convince your staff, then development will be an uphill battle.

ACTION 2: *Create an accessibility policy.* Begin by writing a rough draft yourself. Don't worry too much about specifics at this stage, but concentrate on creating a skeleton for discussion. Next, sit down with your development team and review the current state of the site. In particular, look at the common accessibility issues and identify which ones need to be addressed. Also discuss with your developers what they consider an achievable goal for site accessibility. Finally, work with your content providers on a training program that will ensure accessibility over the long term.

ACTION 3: *Finish what you start.* After you've completed these actions, consider your long-term strategy for ensuring accessibility. Draw up a periodic test plan, preferably using real disabled users. Also ensure that somebody is ultimately responsible for the ongoing accessibility of your site. Ideally, this person should come from your development team and thus understand the technical detail.

I recognize that some of what I've suggested in this chapter may be overkill for smaller organizations, but you should be able to tailor the discussion to your needs. The key is ensuring accessibility over the long term—and to a large extent, that depends on the tools you use. This brings us nicely to content management systems.

website owner's manual

Paul Boag

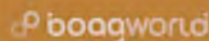
There's a divide in skill and experience between the marketers, managers, and business owners of websites, and the designers and developers who build them. *Website Owner's Manual* is the bridge. From it, site owners and web builders learn how to work together to imagine, create, and maintain a winning web presence.

Website Owner's Manual helps you

- Form a vision for your site
- Select a web design agency
- Establish performance metrics
- Make smart decisions about costs
- Create engaging sites, build your brand

This book is perfect both for site owners and web pros who need to cooperate effectively. It's fully illustrated, totally practical, and fun to read. You'll get a jargon-free overview of web design, including accessibility, usability, and online marketing, along with a practical understanding of the technologies, processes, and ideas that drive an exceptional website.

Through his web design firm Headscape, **Paul Boag** works with business clients to build effective websites. A popular speaker and author, Paul hosts the web's longest running design podcast at boagworld.com.

 boagworld

For online access to the author and a free ebook for owners of this book, go to www.manning.com/WebsiteOwnersManual

FREE EBOOK
SEE INSERT

“Great book! Step-by-step overview of website creation and management.”

—Aleksey Nudelman
C# Computing, LLC

“If a website had a glove box, this manual would be in it.”

—Sheldon Kotyk
TruthMedia Internet Group

“The perfect companion for today's multi-hat web manager.”

—Andy Yeates
Environment Agency

“Every website manager needs this book! Paul is witty and knows what he is talking about.”

—Robert Hanson
author of *GWT in Action*

“The Bible of website management for non-techie.”

—Andrew Grothe
Triware Technologies

ISBN-13: 978-1933988450
ISBN-10: 1933988452

