lucid

Website Accessibility for Developers

Lucid Communications Ltd
Prepared by Neil Turner
January 2006

Document History

Version	Author	Comments
v1.0	Neil Turner	Initial Release

1. Executive Summary

This document gives an overview of the technical considerations in creating websites that meet the W3C Web Content Accessibility Guidelines (WCAG). These guidelines are based around four main principles, which are:

- Separate structure from presentation
- Make content understandable and navigable
- Provide text equivalents of non-textual information
- Do not rely on specific hardware

This is not intended as a comprehensive guide to the W3C standards. Rather, it provides a general overview and highlights a few key points, suggesting solutions.

It is recommended that developers take a pragmatic approach to complying with the W3C guidelines; there is always a trade-off between compliance and attractive design.

2. Web Content Accessibility Guidelines

The World Wide Web Consortium (W3C) publishes a set of guidelines outlining good practice in making websites accessible by those with disability. These are the basis of many governments' regulations regarding accessibility, including the United Kingdom (The Disability Discrimination Act, 1995) and the United States of America (Section 508 of the Rehabilitation Act, amended 1998).

2.1. The Priority System

WCAG contains 14 guidelines, with 65 checkpoints for compliance. Each checkpoint is assigned a "priority" of level 1, 2 or 3. Websites meeting all level 1 checkpoints are considered to hold **WCAG Conformance Level 'A'**. Meeting all level 1 **and** 2 checkpoints makes a site Level 'AA' conformant, while meeting all checkpoints of level 1, 2 and 3 entitles a site to refer to itself as Level 'AAA' conformant.

It is Lucid company policy to develop all websites to at least Level 'A' (and preferably Level 'AA') compliance unless specific functionality is required which could not be made to comply. In our experience, Level 'AAA' compliance is excessive, and makes it extremely difficult to build attractive websites. In fact, even the W3C website is not WCAG Level 'AAA' compliant!

3. The Advantages of Compliance

There are many reasons why making your site accessible can be of advantage to you. You increase your potential user base, improve your search engine ranking, save time and effort on future updates, and avoid potential legal trouble.

3.1. Larger User Base

The first thing to understand is that in the context of the Internet, the idea of a "disabled" person has a far broader scope than its common usage. Anyone who might have difficulty reading from a computer screen (including the colour-blind), and anyone who has trouble operating a mouse or

keyboard is included in the definition. This adds up to around 14% of the population of the United Kingdom. Clearly, it's unwise to exclude such a large group of people.

In addition, it is not only the disabled for whom sites need to be made accessible. Other groups include:

- Those with **slow internet connections**, who may turn off certain features (such as the use of images).
- Those using **old or unusual browsers** that may not render the page exactly as you intended.
- Those using **small screens**, like those on PDAs or mobile phones.
- Those who, for one reason or other are **missing certain hardware** such as a mouse.

Put together, all these users represent a substantial portion of your potential audience. From both a financial and a PR perspective, it's important not to ignore them.

3.2. Freedom from Legal Issues

Section 19(1) of the United Kingdom's Disability Discrimination Act of 1995 states:

> **19.** – (1) It is unlawful for a provider of services to discriminate against a disabled person–
>
> (a) in refusing to provide, or deliberately not providing, to the disabled person any service which he provides, or is prepared to provide, to members of the public;

Non-compliance could therefore end up being extremely expensive in both legal fees and settlement costs.

On the other hand, enforcement of this law is contingent on a disabled member of the public making a claim of discrimination against your site. If you are confident that no such claim will be made (because your site has no disabled users, for example), compliance for legal reasons is not essential. It is perhaps worth noting that as of January 2006 there is only one recorded legal case concerning web accessibility: *Maguire vs Sydney Organising Committee for the Olympic Games,* which was pursued under Australian law in 1999.

3.3. Better Search Engine Ranking

In a lot of ways, a **search engine** is quite like a blind user utilising a screen reader; search engines ignore the layout and graphic design of a page, and simply index the textual content. As a result, websites that are easier for search engines to read tend to be better placed in the rankings than those that aren't.

3.4. Easier Maintenance

In addition, there is a substantial overlap between design techniques that promote accessibility and those that result in a **low-maintenance** site. For example, accessibility places a lot of emphasis on separating the *content* of the site from its *design*, so that screen readers and other limited browsers can read the page. The separation of content from design is also a good idea from a technical perspective, as it enables the webmaster to make updates to the content of the

site without interfering with its design. This generally reduces the time it takes to make any given update, and reduces the cost of maintenance.

Further, conforming to accessibility standards allows you to make use of automatic validation tools like those provided by Microsoft's Visual Studio. This facilitates more efficient debugging and therefore a more rapid development process.

4. How to Build an Accessible Website

The full WCAG checklist is available at http://www.w3.org/TR/WCAG10/full-checklist.html. In addition, there are a number of automated tools that check most (but not all – certain aspects of usability can only be tested through use) of the points for you. One example is the Watchfire tool: http://webxact.watchfire.com/. There are also a number of Firefox extensions designed to check accessibility, such as *TAW3 With a Click*: https://addons.mozilla.org/extensions/moreinfo.php?id=1158&application=firefox.

However, all of these can only check for accessibility problems *after* you've built your website. This can necessitate you going back and redesigning pages you had thought complete. A better approach is to incorporate accessibility into your designs from the beginning.

4.1. Separating Structure from Presentation

A flat HTML page contains three types of information: content, structure and presentation. The content is the actual message being given to the user, the textual and graphic data. Structural information describes how this is laid out; headings, paragraphs, tables and so on. Presentation information describes what it all looks like; fonts, colours, dimensions, positions.

Simple screen readers capture only the content of the page and some structural information, but completely ignore presentation information. To make it easier for these to operate, it is always advisable to put the content in one place and the presentation in another. The best way of doing this is through the use of **Cascading Style Sheets** (CSS), with the .html page containing only content and structure, and a separate .css file containing all of the presentation information.

Another point to bear in mind is that tables should be used as a structural element only, not as a presentation element. In other words, one should only use a table when there is data to be laid out in tabular format. One should *not* use tables to arrange the general layout of a page. Instead, make use of the more flexible tag to divide the page into sections (header, menu, footer, etc), and use CSS to position these sections on the page. This will have the added benefit of allowing you to write your code with the main content at the top of the page (rather than after the header and side navigation bar). This will mean that more of your content will be indexed by search engines, potentially improving the ranking of your site.

Finally, make use of the <h1>, <h2> and <h3> heading tags. These are structural tags intended to help you divide up content. **Do not** use header tags just because you want to have emphasis on a particular line. Pay attention to the nesting of your elements – do not put an <h3> tag directly inside an <h1>, for example.

4.2. Making Content Understandable and Navigable

The W3C recommends that you make use of consistent page layout, commonly used icons and language which is as simple as possible within the constraints of what is appropriate to the content of the site.

Consistency in layout is easily achieved: multiple pages can all refer to the same .css file, so a given element (or class of elements) will look the same regardless of which page it's on.

On a more technical level, it is recommended to make use of shared code segments for the navigation menus on every page of the site. In ASP.NET, this means utilising User Controls. In .NET 2.0, the use of Master Pages and the <asp:Menu> tag make this still easier. If the website you are building does not use server-side scripting, then at least try to ensure that the menu is in the same place and has the same layout on every page.

The use of a breadcrumb trail (e.g. Currently Viewing: Home > Section 1 > Subsection 1.2) can also make navigation much easier for users.

The visually impaired and those who use small screens can't scan the page quickly with their eyes. Instead, these users will often tab from one link to the next or review a list of available links on a page. As a result, it is important that you link phrases – the text that goes between the <a> and tags – make sense when read out of context. If this is not practical, use the "title" attribute of the <a> tag to enter a description of the page to which the link refers.

Finally, avoid the use of frames if at all possible. Frames can cause havoc for screen readers and small screens, and are in any event an outdated and unattractive technology.

4.3. Providing Textual Equivalents

Screen readers have not yet advanced to the point where they can translate the content of an image or a video into audio. In fact, it's unlikely that they ever will. As a result, it falls to the developer to provide textual equivalents of information provided in any other form. This means utilising the *alt* and *longdesc* properties of images, graphical buttons and animated gifs.

When videos, Flash animations or Java applets are used to convey information, there are a few extra considerations:

- You need to provide a voiceover containing the key information that is presented visually.
- Where information is presented by voice, captions (subtitles) are required.
- The audio and video should be properly synchronised so that the user gets the same information at the same time, by video or audio.

When using image maps, you should provide a list of text links for every region of the map. (Note that we at Lucid do not recommend using image maps for navigational purposes, since they tend to violate the principle of keeping navigation simple and consistent.)

4.4. Hardware Independence

Common though the mouse and keyboard are, they are by no means ubiquitous. Many users find them difficult or impossible to use. It is suggested that you bear this in mind when designing your website.

A very common failing is in the use of JavaScript. All developers have at some point made use of the *onclick* property of an element to initialise a JavaScript function. But without a mouse, *onclick* is impossible. You should always use *onkeypress* in conjunction with *onclick*, calling the same function.

In any event, it is not recommended that you use JavaScript for essential functions on your site – many users browse the Internet with script support turned off for security reasons. It is good practice to provide an alternative page which presents the same information without using scripts.

When creating applets (in, say, Java or Flash), ensure that they too can be operated by multiple types of hardware (usually keyboard *and* mouse is sufficient).

4.5. Colour

It is good practice to make your website accessible to the colour-blind. Remember that some colour combinations, particularly greens and yellows, are often indistinguishable to a lot of people. There is an excellent Web Designer's Colour Chart available at http://www.visibone.com/color/chart2x.html, which gives a very effective demonstration of which colours are best used when designing a site with colour-blind users in mind.

However, there are also some users still using devices that do not render in colour at all, so it is usually best practice to ensure that you do not attempt to convey any information through colour alone (e.g. "A green square means success, a red square means failure…"). Also, always ensure that your foreground colour stands out from your background colour sufficiently that it can still be seen on a monochrome monitor.

4.6. Language

You can assist screen readers by indicating the language of a particular piece of text by utilising the *lang* attribute. The attribute can be applied to the whole page (`<html lang="en-gb">`) or to one section (`<div lang="fr">`).

5. The Final Word

The fact is that accessibility requirements are not going to go away. In fact, as the importance of the Internet grows, it is going to become more and more important to provide means for the disabled to use your website. Acknowledging this, the W3C is soon to release version 2.0 of the WCAG, updating the standards to make them more easily machine-testable and to cover newer web technologies. Soon there will be no excuse for not implementing accessibility standards.

With accessibility, as with any other aspect of design, it is far easier to incorporate it into you initial designs than it is to try to tack it on later. It may take a little longer to build your site, but you will save time and effort in the long run.

