Karl Groves

Web Accessibility Viking

The 6 Simplest Web Accessibility Tests Anyone Can Do

Published by Karl Groves on September 5, 2013

What if I told you that the WCAG 2.0 recommendation by the W3C is 36 pages, printed? In addition, "How to Meet WCAG 2.0" is 44 pages and "Understanding WCAG 2.0" 230 pages. Not only that, but the accompanying Techniques and Failures for WCAG 2.0 is 780 pages, printed. There are approximately 400 Techniques and Failures. All of it is wonderful information created by some of the most knowledgeable accessibility folks available. Unfortunately it presents a sort of barrier-to-entry for folks who really just want to know what they need to do to be more accessible. This barrier works both ways: one shouldn't assume this is easy work. You can't just flip a switch and have an accessible site. To truly understand accessibility and accessible web development requires extensive knowledge of HTML, CSS, JavaScript, the DOM & BOM, accessibility APIs, assistive technologies, and how people with disabilities use computers. I've dedicated my career to it and find out the more I know the less I understand.

What about those for whom accessibility isn't their career choice? What about those who don't want to know things like the complicated interplay between markup, DOM interfaces, and accessibility API role mappings? What about people who want to know: "How does my website perform for people with people with disabilities?" Here you go. 6 tests that anyone can do without any development knowledge.

1. Unplug your mouse and/ or turn off your trackpad

Possibly the quickest and easiest way to test your website's accessibility is to unplug your mouse and/ or turn off your track pad. This would require, naturally, that you use only your keyboard to interact with the site. The test process is simple: *Interact with the site using only the keyboard*. The 'Tab' key will allow you to traverse forward in the tab order. Activating the 'Shift' and 'Tab' keys at the same time will traverse backwards in the tab order. Pressing 'Enter' will follow links, and so on.

1. Can you interact with *all* controls, links, and menus using only the keyboard?
2. Can you *see* what item has focus at all times?
3. Does the visual focus order match the intended interaction order?

What does this mean?

A large number of persons with disabilities cannot use a mouse to interact with the web. Users who are blind can't see the mouse pointer, users who are low vision may prefer the keyboard, users with motor control disorders may be unable to use a mouse, and it may be painful for those with chronic pain to use a mouse. If you cannot answer 'Yes' to all the above questions, then the site requires repair in order to be accessible. Keyboard accessibility is required for users who are blind, low-vision, or who have motor control disorders. Keep in mind: the visual focus indication is especially important for general usability and critical for users with low vision or who have motor control disorders. Not all users with disabilities are blind!

2. Turn on High Contrast Mode

In the Windows operating system, High Contrast Mode allows Low Vision users, users with light sensitivity, and sometimes users with Dyslexia a convenient means of improving their ability to successfully use the computer. Windows High Contrast Mode changes the foreground and background colors to create higher contrast. Colors on the site are essentially removed entirely. All background is black and all foreground text is a significantly brighter color such as white or yellow (users can customize this). With High Contrast Mode turned on, interact with the site.

What does this mean?

According to WebAIM's Survey of Users with Low Vision, 30% of respondents state they used High Contrast Modes. One of the best articles out there on this topic is ~~Techniques for High-Contrast-Friendly Icons~~ by Todd Kloots. In terms of sheer numbers, there are far more people with low-vision than those who are totally blind, so if your accessibility strategy is built around whether the site works in screen readers, you may want to reevaluate that approach.

3. Turn off Images

To turn off images in FireFox: go to Tools -> Options -> Content -> and then uncheck the checkbox labeled "Load Images Automatically" -> then select the OK button. **In Internet Explorer:** go to Tools -> Internet Options ->Advanced -> and then uncheck the checkbox labeled "Show Pictures" -> and activate the OK button. Other browsers are similar.
With images turned off:

1. Does the content make sense?
2. Does the content become harder to understand?
3. Is any content now missing?
4. Do any important controls disappear?

What does this mean?

Images are good for usability and accessibility. Users with cognitive disorders can benefit greatly from the combination of images and text to understand information. That said, images shouldn't be *required* to understand the page and shouldn't be relied upon for important UI controls. Any text presented in images should be placed in the markup instead, except in the case of branded content like logos or trademarks. Any images which present important content should be given a text alternative and some images are best described directly in the content of the page.

4. Check for Captions or Transcripts

OK, I lied about which of these tests was the easiest. This one totally is easier than anything you can do. If you have media on your site, check for captions, transcripts, and other possible alternatives. Wherever you have media:

- Are there captions on the video directly or is there a control in the player that turns on/ off captions?
- Is there an alternative version with audio description or a control in the player that turns on/ off audio description?
- For videos with a lot of dialog, is there a text transcript on the page or link close to the video player that goes to a transcript?

Other requirements exist for video-only and audio-only scenarios, but the above will cover most media on most sites.

What does this mean?

Its easy to assume that media accessibility only impacts users who are blind or deaf, but in fact accessible media alternatives also benefit users with low vision, users who are hard of hearing, and users with cognitive disabilities. Ostensibly a text transcript is good for SEO but also beneficial for users who wish to be able to search for a specific bit of content in the media. Note: not only should the media content itself be accessible but so should the player. Look for a future blog post on media player accessibility.

5. Click on Field Labels

Forms are the top method of interacting with websites and the top method of measuring conversion. Unfortunately, by volume, forms-related accessibility problems are among the most frequent issues. Issues with forms tend to fall into three main categories: Missing or incomplete labeling, ineffective error handling, and poor focus control. One extra-simple way to test for the existence of form labels is to simply click on the text label adjacent to the field.

- When you click on the label next to a text input or textarea, does the cursor go into the field?
- When you click on the label next to a radio button or checkbox does that select the adjacent option?
- When you click on the label next to a `SELECT` element, does that place focus on the `SELECT` ?

What does this mean?

There should be a one-to-one relationship between the label and the control. Unlabeled or improperly labeled controls must be fixed. There's a lot more to consider when considering the accessibility of forms, but this quick check will at least let you tell if you've met a very important accessibility criteria: proper labeling of forms. Without properly labeled fields, none of the other forms best practices really matter.

6. Turn off CSS

Cascading Stylesheets is the preferred method of handling layout and visual presentation of web interfaces. Using CSS properly reduces document weight and increases flexibility for users. At this point that's sort of old news. From an accessibility standpoint, there are a few things to look out for.
To turn off CSS in FireFox, go to View -> Page Style -> and then select No Style. Doing so in Internet Explorer is also very similar. Go to View -> Style -> and then select No Style. Other browsers are similar as well. After turning off styles check the following:

- Background images will disappear. Did important content disappear or did any controls, icons, or other actionable elements disappear?
- Is it now difficult to understand things like error messages or other previously-visual cues?
- Is content still displayed in a reasonable, easy to understand order?

- Does any color, including background color remain?
- Do any text presentational styles remain?

What does this mean?

One of the biggest CSS-related issues I find in testing these days is the use of background images for content or controls. So, if controls or content completely disappear after turning off images, this must be repaired. If any other visual presentation remains with styles turned off, that means that old, deprecated, presentational HTML is still in use. This will mean that these presentational methods cannot be overridden by users who've created custom style sheets. According to WebAIM's Survey of Users with Low Vision, 19% of users with Low Vision users browse the web with "Customized page colors or custom style sheets".

Bonus: Diagnostic.css

Diagnostic.css is a CSS (Cascading Stylesheets) file which, when applied to a web page, will highlight accessibility errors in the page. It allows testers to do a super-quick automated test on web pages without the need to buy or install a tool.

Just. Use. It. Yourself.

I test websites, web applications, software, and mobile/ tablet apps for accessibility almost all day for my day job. One of the best pieces of advice I can give is to just use the system yourself. Put yourself in the role of a user of your system and actually use it. You'll be amazed at the insight you'll get into your system's quality.

Do you need help with accessibility? Grab a time on my calendar for a quick chat!

Published in Accessibility Testing

Previous Post
CSS generated content is not content

Next Post
Web Accessibility Testing Tools: Who tests the DOM?