

HONORS DISTINCTION PROJECT

by

Laura Davis

Submitted to the Honors College
Missouri State University

December 2018

Approved:

Dr. Sean Herring, Project Director

Dr. W.D. Blackmon, English Department Head

Dr. John Chuchiak, Director of the Honors College

**WEB USABILITY GUIDELINES FOR TECHNICAL WRITERS: VISUAL
ACCESSIBILITY**

An Honors Distinction Project

Presented to

the Department of English

and the Honors College

Missouri State University

In Partial Fulfillment

of the Requirements for

Distinction in the Major in Professional Writing

by

Laura Davis

December 2018

WEB USABILITY GUIDELINES FOR TECHNICAL WRITERS: VISUAL ACCESSIBILITY

Department of English

Missouri State University, December 2018

Laura Davis

ABSTRACT

The work that technical writers perform is increasingly shifting toward the creation of electronic content, specifically websites. Web usability research, however, has declined significantly over the past two decades as more web designers rely on web analytics tools. While web analytics tools can be useful in collecting and evaluating quantitative data, for revealing and predicting website users' browsing habits, an overreliance on those tools has created gaps in qualitative aspects of website design. Such gaps include web heuristics principles and accessibility standards. In this research, I have focused on web usability, specifically regarding accessible web design as it applies to users with visual impairments. I demonstrate two methods to make websites more accessible for users with visual impairments: Form design for people who use screen readers and high-contrast theme switchers for people with contrast perception difficulties. In each example, I make some basic modifications to a web page and present them in a Before and After Demonstration (BAD) to establish the effectiveness of making simple accessibility changes to websites to benefit people with visual impairments. At its core, this research is intended to help technical writers integrate accessibility into their designs and to encourage an increase in heuristics-based website usability research and advocacy among technical writers who design websites. As user-advocates, technical writers have the responsibility to ensure that accessibility principles make their way to becoming industry best practices on par with writing well and content design.

KEYWORDS: Web usability, web accessibility, web usability heuristics, web design, technical communication

**WEB USABILITY GUIDELINES FOR TECHNICAL WRITERS: VISUAL
ACCESSIBILITY**

by

Laura Davis

An Honors Distinction Project
Submitted to the Honors College
Missouri State University
in Partial Fulfillment of the Requirements
for Distinction in the Major in Professional Writing

December 2018

Approved:

Dr. Sean Herring, Project Director

Dr. W.D. Blackmon, English Department
Head

Dr. John Chuchiak, Director of the Honors
College

TABLE OF CONTENTS

INTRODUCTION	1
LITERATURE REVIEW	4
Making websites visually accessible	6
Designing websites for people who use screen readers and mobile assistants	7
Designing websites for people with contrast perception impairments.....	11
METHODS	12
Choosing accessibility features in need of standardized implementation.....	12
Implementing accessibility features	13
Demonstrating the usefulness of accessibility features	14
Testing accessibility features with assistive tools.....	15
FINDINGS	16
Benefits and difficulties of implementing visual accessibility features	16
DISCUSSION	19
Web accessibility and qualitative design	19
Refutation of arguments against the implementation of accessibility features	21
CONCLUSION	25
OTHER RESOURCES	26
REFERENCES	27
APPENDICES	30
Appendix A: LinkedIn mobile mockup before	30
Appendix B: LinkedIn mobile mockup after	30
Appendix C: LinkedIn form source code before	31

Appendix D: LinkedIn form source code after	31
Appendix E: Wikipedia mockup before	33
Appendix F: Wikipedia mockup after – high luminance (standard theme)	34
Appendix G: Wikipedia mockup after – low luminance (dark theme)	35
Appendix H: Wikipedia HTML and JavaScript changes.....	35
Appendix I: Wikipedia CSS changes.....	35

LIST OF FIGURES

Figure 1: The LinkedIn mobile Sign in page.	16
Figure 2: LinkedIn mobile login page before changes.	30
Figure 3: LinkedIn mobile login page after changes	31
Figure 4: Wikipedia page mockup before changes	34
Figure 5: Wikipedia page mockup after changes with high luminance theme	35
Figure 6: Wikipedia page mockup after changes with low luminance theme.	35

KEY TERMS

A11y – An abbreviated form of the word “accessibility” (“a” and “y” with 11 characters in between) commonly used within the technology industry to refer to accessibility issues, especially regarding accessibility on computer systems, such as user interfaces and web pages.

See also: *Web accessibility* and *Web usability*.

Assistive technology – A device or program that helps people with disabilities perform specific tasks. In this context, this includes technologies that help users with disabilities use computer systems.

See also: *Mobile assistant* and *Screen reader*.

ADA – An acronym for the Americans with Disabilities Act of 1990, which states that people cannot be discriminated against because of their disability status and that people with disabilities are entitled to workplace and public accommodations based on accessibility requirements established by Federal and state governments (42 U.S.C. § 12101). ADA regulations are only enforced for websites created and maintained by the Federal Government.

See also: *Section 508*.

BAD – Before and After Demonstration. A method of instruction commonly used in teaching good web design practices, especially usability, in which the demonstrator creates a Before version of a web page and an After version to demonstrate the effectiveness of a specific design principle or usability implementation.

Contrast ratio – Short for “luminance contrast ratio.” The contrast ratio of a web page determines how well most users can read text on a page.

See also: *High luminance* and *Low luminance*.

High luminance – A text-background theme in which the dark text is presented on a light background. This effect is sometimes called a “light theme” or a “standard theme”.

See also: *Contrast ratio* and *low luminance*.

Low luminance – A text-background theme in which light text is presented on a dark background. This effect is sometimes called a “dark theme” or an “inverted theme”.

See also: *Contrast ratio* and *high luminance*.

Mobile assistant – An assistive technology that reads the content of a mobile device’s screen, similar to a screen reader. These technologies are often applications that are included with a mobile device’s operating system. Examples include Voice Assistant and TalkBack for Android (depending on operating system version) and VoiceOver for iOS.

See also: *Assistive technology* and *Screen reader*.

NVDA – An acronym for Non-Visual Desktop Access, a free, open-source screen reader for Windows.

See also: *Screen reader*.

Screen reader – An assistive technology that reads content on an electronic device to a user. Screen readers are commonly used by people with visual impairments, but also by people with learning disabilities like dyslexia. Popular screen readers include JAWS and NVDA.

See also: *Assistive technology* and *NVDA*

Section 508 – Refers to the Section 508 Amendment to the Rehabilitation Act of 1978, which applies accessibility regulations specifically to information technology and computer systems (29 U.S.C. § 794d). Like the ADA, Section 508 regulations are only enforced for websites created and maintained by the Federal Government.

See also: *ADA*.

SEO – Search Engine Optimization. A marketing strategy that improves the “visibility” of a web page for web crawlers, such as those created for indexing and searching the Internet. SEO directly affects a web page’s rank and therefore is integral in maximizing a website’s revenue.

Universal design – A school of thought in web design which states that all websites should use a set of principles similar to Nielsen’s 10 usability heuristics that maximizes usability and accessibility for as many users as possible. Universal design is also known as Design for All.

See also: *Web accessibility*, *Web usability*, and *Web usability heuristics*.

W3C – Acronym for the World Wide Web Consortium, an organization founded in 1994 by the inventor of the World Wide Web, Tim Berners-Lee. The primary goal of the W3C with the goal of creating standardized web design guidelines that make the World Wide Web usable and accessible to all people (W3C).

Web accessibility – A principle of web design which states that websites should be made as accessible as possible to people with disabilities. Although accessibility is typically treated as a separate issue in web design, it falls under the large umbrella of usability.

See also: *Ally*, *ADA*, *Universal design*, and *W3C*.

Web usability – A principle of web design which states that websites should be made as easy to use as possible for all website users. Includes areas such as: Page design, content design, information architecture, accessibility, and internationalization.

See also: *Web Accessibility* and *Web usability heuristics*.

Web usability heuristics – Broad guidelines that refer to specific user-focused aspects of web design. The term was created by usability expert Jakob Nielsen in his seminal work on web usability published in 1994: “Ten usability heuristics for User Interface Design.”

See also: *Web usability* and *Web accessibility*.

INTRODUCTION

As more of the work technical writers perform continues to shift toward creating electronic content, specifically websites, the usability of websites falls within the purview of technical writing because work in the field is often user-centric. I have focused my research on web usability, specifically regarding accessibility standards, to locate gaps in both research and practice related to the heuristics of website usability. This research is intended to help technical writers integrate accessibility principles into their designs and to encourage technical writers to have an increase in focus on the usability of heuristics-based website design.

Until recently, web design has relied on usability heuristics researched extensively in the 1990s and early 2000s, which focuses on user-centric, or human-based, web design. However, after those decades, research on usability heuristics has declined sharply in part due to a shift in favor of quantitative measurements collected with tools like web analytics. Additionally, few studies regarding web usability, especially usability heuristics, have been conducted in the field of technical communication over the past 10 years. While web analytics tools can be useful in collecting and evaluating quantitative data for revealing or predicting website users' browsing habits, an overreliance on those tools has created gaps in qualitative aspects of website design, such as heuristics principles and accessibility standards.

As technical writing continues to shift toward creating and maintaining web-based content, it requires the application of up-to-date web usability guidelines. Many of those updates can be accomplished by adhering to guidelines created by the World Wide Web

Consortium (W3C), which consists of approximately 500-member organizations from around the world. Since October 1994, W3C continues to be the leader in establishing and maintaining web-usability in such areas as layouts, navigation, and accessibility, that ensure meeting industry standards.

Technical writers and web designers may benefit from my qualitative analysis of web usability to ensure that those in the field continue to communicate effectively for a rapidly growing audience using and accessing websites. After concluding that qualitative website usability research had declined over the past two decades, I focused my research on identifying areas where usability heuristics had become outdated and were no longer being used in website design. During my research, I sought to answer the following questions about website usability:

1. What can technical writers do to make websites more accessible and accommodate the needs of many different users?
2. Is there a consistent absence of usability-focused web design principles in contemporary web design? If so, how widespread are these problems and are they easy and inexpensive to solve?
3. Why are qualitative website usability research and the establishment of consistent design with guidelines important?

The first two of these questions are answered primarily with research conducted during my literature review. The third question is answered in examples of web pages from well-known websites that do not adhere to specific accessibility guidelines. My redesign of

those well-known websites demonstrates the ease with which designers can solve these issues and make their websites more usable.

Although there are many areas of accessibility, that assist people with visual, auditory, speech, motor, and cognitive disabilities (Jakob Nielsen, 2000), I have narrowed the focus of this paper to address visual accessibility. It is one of the largest accessibility areas with serious need for improvement (pp. 302-311).

LITERATURE REVIEW

In order to answer my first question regarding research gaps in web usability guidelines, I conducted a literature review and determined that the most gaps existed with regard to website accessibility. During preliminary research, I located more than 30 sources to provide secondary evidence regarding the establishment of web usability heuristics; however, a large portion of these sources were older than 10 years and some were more than 20 years old.

Although sources equal to or less than 10 years old may be acceptable in academic research, the rapid evolution of technology results in technical concepts and standards becoming obsolete in less than 10 years. Hence, my approach excluded all sources equal to or greater than a decade old, except for seminal texts on web design and usability heuristics authored by Jakob Nielsen (1994), the originator of web usability heuristics, and the original comprehensive web usability guide published by the Department of Health and Human Services, all of which were published in the 1990s or early 2000s.

Nielsen's usability heuristics, in particular, were a starting point for my research and helped me focus on a handful of heuristics that include web accessibility. In "10 Usability Heuristics" Nielsen (1994) lists the following broad guidelines:

1. **Visibility of system status:** Ensuring the users are informed about what's going on
2. **Match between system and the real world:** Speaking the users' language and using familiar concepts

3. **User control and freedom:** Supporting common user actions
4. **Consistency and standards:** Being consistent in writing and following established conventions
5. **Error prevention:** Prevent errors by eliminating the possibility for the user to make mistakes
6. **Recognition rather than recall:** Reduce users' cognitive load
7. **Flexibility and efficiency of use:** Cater information to users at different levels
8. **Aesthetic and minimalist design:** Eliminate extraneous information and graphics
9. **Help users recognize, diagnose, and recover from errors:** Use plain language in error dialogs and troubleshooting instructions
10. **Help and documentation:** Always provide concise and useful help or documentation where possible

Out of those 10 broad heuristics, I focused primarily on the following three areas: “3. User control and freedom,” and “7. Flexibility and efficiency of use,” and “8. Aesthetic and minimalist design.” All of those heuristics relate strongly to visual accessibility in web design.

Further, I have categorized those three heuristics into a handful of accessibility areas that relate to visual accessibility. The visual accessibility category includes conditions such as impaired vision, low-vision, and blindness, along with visual impairments that affect contrast, depth, and color perception. Further, visual accessibility

is also beneficial for people with learning disorders, especially dyslexia, who also benefit from assistive technologies like screen readers. Of Nielsen's heuristics, "User control and freedom," and "Flexibility and efficiency of use," and "Aesthetic and minimalist design" apply to visual accessibility.

In the following sections, I have grouped literature according visual accessibility guidelines that apply specifically to web design for screen readers and mobile assistants, as well as guidelines that apply to contrast, in order to answer my first question: "What can technical writers do to make websites more accessible and accommodate the needs of many different users" with an emphasis on visual accessibility?

Making websites visually accessible

Concerning web usability, visual accessibility has long been an area of intense focus for web accessibility advocates. Websites not being accessible for visually impaired users is still a common source of frustration, especially for those who rely on assistive technologies like screen readers, because accessibility is often treated as an afterthought in web design rather than a priority. Although many of these visual accessibility standards have existed for nearly two decades, the rate of adoption in design is still relatively low and therefore still requires the attention of web designers.

In the following sections, I discuss methods of designing websites to be more visually accessible, especially in areas that apply to people who use screen readers and mobile assistants, as well as those who require theme modifications for contrast adjustments on web pages.

Designing websites for people who use screen readers and mobile assistants

In a 2007 study, researchers Jonathan Lazar, Aaron Allen, Jason Kleinman, and Chris Malarkey at the Universal Usability Laboratory at Towson University discovered that alt text, specifically missing, misleading, and non-descriptive alternative text is an everyday frustration for people who use screen readers (pp. 260-261).

Since the first public draft of the W3C's Web Content Accessibility Guidelines (WCAG 1.0) was published in 1998, accessibility advocates have recommended including text alternatives for non-textual content using the `alt` attribute. Specifically, the W3C (2018b) recommends that "All non-text content that is presented to the user has a text alternative that serves the equivalent purpose" with a handful of exceptions, such as graphics that are for decoration only (WCAG 2 § 1.1.1).

Similarly, usability researcher Jakob Nielsen (2000) asserts that other non-textual content, such as bullet points, require no alternative text because using a description such as "large blue bullet" would provide no value to people using screen readers. Instead, Nielsen recommends using a null, or empty, alt description (such as `alt=""`) in these situations (pp. 305-306). Nielsen also argues, however, that alt text should not be overly descriptive (p. 305). In this case, writing good alternative text is equally important as including it in a page's design. The W3C (2017) asserts that, although some users prefer detailed alternative descriptions while others prefer short descriptions, alt text as a rule "should convey the same meaning as the image" and that users should be able to "get the important information from the image in the alternative text" ("Image text alternatives").

Further, the W3C (2017) notes that the appropriateness of alternative text depends on the context and the purpose of the website (“Image text alternatives”). For example, on one hand, a website for a local animal shelter with an image of a woman holding a dog might require alternative text depending on the use of the image. If the image was simply a stock photo that does not convey information to users, it would require null alt text. On the other hand, if the image were part of an article on local residents who have adopted pets from the shelter, it would require alt text, especially if the image is referenced in the article text itself.

Rather than using non-descriptive alt text like “A woman holding a dog,” or an ineffective visual description like “A tall woman with long hair holding a medium-sized brown dog,” the description should convey information that relates to the article itself, such as “Nancy holding Rufus, a dog that she adopted from our shelter in March 2018.” Therefore, when it comes to writing alt text, we should remember the technical writing principle which teaches us that our users’ time is valuable and that we should always strive to create clear, concise, and informative writing wherever possible.

Further, text alternatives are important for users specifically where diagrams and illustrations are concerned. Since the purpose of diagrams and illustrations are to convey important information to readers, providing an alternative, detailed description of the content of diagrams and illustrations is as important for people who use screen readers as describing other non-textual elements like images. In *Developing quality technical information: A handbook for writers and editors Third Edition* (2014), authors Michelle Carey, Moira McFadden Lanyi, Dierdre Longo, Eric Radzinski, Shannon Rouiller, and

Elizabeth Wilde argue that “If your text doesn’t sufficiently cover the same information that’s conveyed in the graphic, you must provide an explanation of the graphic in an alternative text element to ensure that visually impaired users have access to equally complete information” (p. 478).

Similarly, the prominent usability specialist and author Janice (Ginny) Redish (2012) concurs that graphics and illustrations should include alternative descriptions, and advocates that such descriptions be placed in the `alt` attribute within the `` element (p. 286). The W3C (2017) argues, however, that “If the image has complex information – such as charts or graphs – the image should have a short alt text to identify the image, and then the detailed description of the information should be provided elsewhere (for example, in a data table)” (“Image text alternatives”). Although either method would certainly be better than not including alt text at all, technical writers must remember that alt text descriptions should be concise; however, there are cases in which complex diagrams and illustrations might need to be longer and more detailed in descriptions that adhere to the W3C’s recommendation.

Further, the U.S. Department of Health and Human Services (2006), who authored the comprehensive *Research-based web design and usability guidelines* recommends that designers should “Provide text equivalents for non-text elements” and “Design forms for users using assistive technologies” (pp. 25; 23). It is notable that, according to the HHS usability handbook, all Federal Government websites are required to comply with the American Disabilities Act (ADA) and Section 508, both of which protect people with disabilities from discrimination and states that people with disabilities

are entitled to workplace and public accommodations, including measures that make websites accessible (p. 22).

Aside from providing alternative text for non-textual content, Lazar et al. (2007) assert that unlabeled input fields are a frequent issue for people who use screen readers or mobile assistants. In fact, the results of the study conducted by Lazar et al. (2007) reveal that poorly designed forms and unlabeled input fields are a source of frustration for visually impaired users on par with missing, misleading, and non-descriptive alt text and similar design issues (p. 257-263).

The W3C suggests labeling interactive page elements, which includes form fields, in their most recent revision of their Web Content Accessibility Guidelines (WCAG 2.1). Concerning the labeling of form fields specifically, the W3C (2018a) recommends that designers should “Ensure that all fields have a descriptive label adjacent to the field. For left-to-right languages, labels are usually positioned to the left or above the field, except for checkboxes and radio buttons where they are usually to the right” (“Ensure that form elements include clearly associated labels”). Although the W3C considers the clear and effective labeling of interactive elements as primarily being a cognitive accessibility issue, especially for users with dyslexia, research conducted by Lazar et al. (2007) demonstrates that form field labeling is equally important for visually impaired users.

Furthermore, people with disabilities that affect reading and comprehension, such as dyslexia, sometimes also use screen readers to improve the accessibility of web content.

Although this guideline has not been as prominent as other accessibility guidelines, such as those concerning alt text, over the past two decades, providing labels for interactive elements is an important aspect of visually accessible web design.

Designing websites for people with contrast perception impairments

For users with visual impairments, that are not addressed with the use of screen readers, providing options to change the color of pages or increase the contrast ratio is an important step in designing more accessible websites.

Redish (2012) argues that web designers should allow users to make changes to their websites, such as text size and contrast (pp. 48-49). Similarly, the W3C (2018b) asserts that background and text colors should maintain a minimum contrast ratio of 4.5:1 (WCAG 2 § 1.4.3). The W3C (2017) also argues that, while high luminance themes (dark text on a light background) is beneficial for many users with visual impairments, low luminance themes (light text on a dark background) are also beneficial for users with contrast perception impairments and people with dyslexia (“Contrast ratio”). Further, the W3C (2017) recommends that “Web browsers should allow people to change the color of text and background, and web pages need to work when people change colors” (“Contrast ratio”). Although design experts, including Redish (2012, pp. 52-53), caution against using light text on dark backgrounds, providing users with the option of changing the look of the page to increase contrast is an important aspect of designing websites that are more visually accessible.

METHODS

In this project, my first method of research was a literature review analysis of the question: “What can technical writers do to make websites more accessible and accommodate the needs of many different users” with an emphasis on visual accessibility? From those findings, my second research question is addressed directly within this “Methods” section.

The goal of this research is to determine which web accessibility features are most in need of standardized implementation, the benefit and difficulty of implementing those accessibility features, and the best methods with which to demonstrate and test the usefulness of said accessibility features. In this section and the following one, I answer my second research question: Is there a consistent absence of usability-focused web design principles in contemporary web design? If so, how widespread are these problems and are they easy and inexpensive to solve?

Choosing accessibility features in need of standardized implementation

While researching gaps in web usability and accessibility issues, I determined that one of the areas most in need of improvement involves visual accessibility. Although visual accessibility issues have been at the forefront of guidelines regarding web usability, I determined that many guidelines, including basic ones like using alt text, are still not being implemented in web design, even on popular sites.

Further, I determined that two of the simplest and most effective ways to increase web accessibility for users with visual impairments is to make websites easier to use with screen readers and to make content easier to read with simple contrast-related style

changes. Improving websites by using alt text for visual elements, properly labeling form fields, and providing users with options to increase page contrast with a theme switcher, may extend making the web more accessible for users with visual impairments if such methods were implemented and encouraged as best practices in web design.

Implementing accessibility features

In order to demonstrate the simplicity and effectiveness of implementing these simple web design features, I chose to find websites that lack accessible features regarding the following areas of focus:

1. Alt text for visual elements
2. Form field labels
3. Background and text contrast

Concerning alt text, my research confirms that social media websites frequently do not implement alt text or allow users to provide alt text manually. The U.S. General Services Administration (GSA) (2017) points out that 84% of employers use social media as a recruiting tool, but that inaccessible social media websites put people with disabilities at a disadvantage (“Why is the accessibility of social media so important”). The GSA also points out that, while some social media sites like Facebook allow users to add alt text to their images, sites like Instagram do not allow alt text, nor do they allow users to implement closed captioning in their videos. Instead of using alt text on Instagram, the GSA recommends putting a detailed description of the image in the picture’s caption (“Tips for making Instagram posts accessible”).

Concerning form field labels, I discovered that some websites, especially mobile versions, do not properly label form fields for use with screen readers. The LinkedIn mobile website is one such example. In this case, markup errors along with the use of HTML values and JavaScript that does not conform to industry best practices creates non-fatal errors in the page that makes reading the login form with a screen reader difficult and, at times, causes the screen reader to not read the form fields at all.

Concerning accessibility with regard to background and text contrast, I determined that the popular site Wikipedia would benefit most from integrating a theme switcher into the site's design. Wikipedia uses a simple high luminance theme by default, which is easy to read for many users; however, users who have difficulty with contrast perception or people with dyslexia would greatly benefit from the implementation of a low luminance theme option. Although Wikipedia does allow registered users to modify the look of the site with Hypertext Markup Language (HTML) and Cascading Style Sheets (CSS), integrating a theme switcher into the page would provide a more accessible alternative for users with disabilities who do not want to create a profile or spend a lot of time modifying their profile's HTML and CSS code.

Demonstrating the usefulness of accessibility features

After determining my area of focus regarding website usability and accessibility issues and determining sites to use as examples for implementing visual accessibility features, I chose to demonstrate the effectiveness of these features with a Before and After Demonstration (BAD) for each page. BADs are an effective tool that is used commonly in web design instructions and are also used by organizations like the W3C to

demonstrate accessibility practices in action. Because of the way modern websites are dynamically served and changed frequently, I determined that it would be best to create mockups of the sites from their source code for the Before example of the BAD, followed by a modified version to demonstrate the effectiveness of accessibility features for the After example of the BAD. Further, in order to code the websites used in the BAD, I utilized HTML5, CSS3, and JavaScript, coding each page by hand in the Brackets text editor.

Testing accessibility features with assistive tools

To test the effectiveness of my changes to each example web page, I primarily used two technologies to test the Before and After versions of the pages for people who use mobile assistants or screen readers.

For the mobile LinkedIn example, I used the built-in Voice Assistant app on an Android (6.0 Marshmallow) device. I also used NVDA to test the page's form fields because it is much easier to use than Voice Assistant in testing and because a desktop computer can run a mobile version of a web page for testing.

Concerning the contrast switcher, it is difficult to test the effectiveness of contrast without conducting usability testing with users who have visual deficiencies which affect contrast perception or reading; however, the W3C (2016) provides a formula to determine a contrast ratio of any two colors.

FINDINGS

After building and testing the example websites, I have found that implementing alt text, form field labels, and contrast options greatly improve the accessibility of each of these pages. While coding and testing each of these BAD scenarios, I attempted to gauge the potential benefit and the difficulty of implementing each of these accessibility features.

Benefits and difficulties of implementing visual accessibility features

LinkedIn form fields. On the mobile LinkedIn webpage, the login form was moderately inaccessible to users who use screen readers. The primary issue that created this accessibility problem was the misuse of `<label>` elements on the form. In this case, the form did include labels, but the designer used them as manual placeholder text instead of using the `placeholder` attribute for those values. Although this unconventional method is not technically inaccessible to screen readers, non-fatal markup errors, including attributes without spaces between them, caused a rendering error that sometimes made it difficult for the screen reader to identify all the form fields. Secondly, the designer gave the email field autofocus, which would be confusing for someone using a screen reader as it is an

Figure 1: The LinkedIn mobile Sign in page.

industry standard to give the top or first field autofocus. Giving other fields autofocus interferes especially with screen readers that jump to the focused part of the page to begin reading. In this case, a user who is using a screen reader might mistakenly believe that they are at the top of the form, which will cause them to avoid entering their first and last name and then cause a form submission error when the user submits the form, leading possibly to a lot of frustration for a screen reader user.

To make this page more accessible, I inserted labels for each field using the `aria-label` attribute, which is well-supported by screen readers. I also kept the placeholder text for the aid of sighted users, but it is important to note that the W3C (2017b) advocates against relying on placeholder text for accessibility because screen readers do not treat placeholders as text. Creating either traditional labels with the `<label>` element or using `aria` attributes are the most accessible practices for labeling form fields. After repairing the labels, I also gave the first field on the page (first name) autofocus to ensure that anyone using a screen reader will be directed to the top of the form, which will help avoid input errors and frustration created by rejected form submissions.

Overall, the difficulty of implementing these features was minimal. Most of the difficulty that I had was in attempting to debug the original page's source code in order to figure out why a form that was, in fact, using labels was still inaccessible to a screen reader. After figuring out the problems, fixing them required no more than a few modifications to the form's HTML attributes.

For more details, including screenshots and code samples, see [Appendices A-D](#) or the [LinkedIn before](#) and [LinkedIn after](#) pages.

Wikipedia theme switcher. Although Wikipedia has a relatively clean, high contrast layout, it is difficult for some users to read long articles on the site with the traditional high luminance theme. The option to switch the page to a low luminance theme would increase the accessibility of the site, especially for readers with difficulties perceiving contrast, people with low-vision, and people with dyslexia.

To make this page more accessible, I created an alternate, low luminance theme in the page's stylesheet. I also inserted a button at the top of the page so users could quickly switch between the two themes. By attaching a JavaScript listener to the theme switcher button, I was able to switch the page's theme from light to dark by changing the CSS class when the user clicked the button.

Concerning the difficulty of implementing this accessibility feature, creating a style switcher was somewhat more complicated than labeling forms. Completing, however, a handful of simple HTML, CSS, and JavaScript modifications is well within the purview of technical writers who have learned basic web design. Further, these basic style changes have the potential to benefit many people, especially on a site as globally popular as Wikipedia.

For more details, including screenshots and code samples, see the [Appendices E-I](#) or the [Wikipedia before](#) and [Wikipedia after](#) pages.

DISCUSSION

One of the essential functions of a technical writer is to ensure that our end product is effective in communication and design. If we are not integrating accessibility into our designs, then we are not only failing to advocate for a large segment of our users, but we are also contributing to the alienation of a large portion of our audience, specifically people with disabilities. In the previous sections, I have demonstrated the high benefits and low difficulty of implementing visual accessibility features which have the potential to improve web accessibility for users with visual impairments. In this section, I answer my third and final research question: Why are qualitative website usability research and the establishment of consistent design with guidelines important?

Web accessibility and qualitative design

A 2016 study by the Pew Research Center (Anderson and Perrin 2017) found that people with disabilities are far less likely to use technology, specifically that only 50% of people with disabilities use technology daily, as compared to 80% of people without disabilities. Further, the Pew Research Center (Anderson and Perrin 2017) found that only about 40% of people with disabilities feel confident using technology. The Pew Research Center (Anderson and Perrin 2017) ties lower rates of technology adoption and confidence in using technology to the generally inaccessible state of many websites.

Tim Berners-Lee, director of the W3C and inventor of the World Wide Web states that “The power of the Web is in its universality. Access by everyone is an essential aspect” (W3C 2018c, “Accessibility in context”). Further, the W3C (2018c) argues that “The Web is an increasingly important resource in many aspects of life:

education, employment, government, commerce, health care, recreation, and more. It is essential that the Web be accessible in order to provide equal access and equal opportunity to people with diverse abilities” and Lee points out that the UN considers access to technology a basic human right (“Accessibility is important for individuals, businesses, society”). In treating web accessibility issues as an afterthought and relying too heavily on quantitative data to make design decisions, web designers have hindered accessibility to their products and services and have thereby impaired the universality and potential of contemporary technology.

In order to improve the accessibility of websites, web designers should rely not only on design based on quantitative tools and data, but also on qualitative principles. Integrating qualitative design guided by usability heuristics and guidelines, like those set forth by the W3C, into web design best practices is vital to ensure that technical writers are serving as much of their audience as possible. G.G. Chowdhury and Sudatta Chowdhury (2011) point out that designers should keep in mind that quantitative research generates data sets that, while useful for giving designers ideas about a set of problems or common usage patterns for technologies, they are insufficient alone and should be combined with qualitative data that “offers detailed information about why people like or dislike certain features and functions of a product” (p. 144).

When designing and testing websites, it is important that designers rely on both tools and more traditional testing methods, like usability testing, in order to ensure the accessibility of their websites. As Linn Steen-Hansen and Siri Fagernes (2016) point out, “automatic testing is not sufficient to ensure accessibility” and that manual heuristic

evaluations are still necessary to ensure that websites are accessible (p. 445). Further, Redish (2012) and the W3C (2018d) advocate for including people with disabilities in testing and usability studies, while also keeping in mind that designers should not assume “that input from one person with a disability applies to all people with disabilities” (p. 311; “Basics”). Redish (2012) also argues that designers should make people with disabilities members of the design process from the beginning and advocates that designers should consider creating personas with disabilities as the beginning of their site design process (p. 32). Similarly, Steen-Hansen and Fagernes (2016) argue that designers should test their own website designs with the tools that they have available to them, such as screen readers (pp. 445-446). Altogether, it is vital that designers make their websites accessible by performing quantitative research, especially by including people with disabilities throughout the design process and by testing their websites with common, widely-available technologies.

Refutation of arguments against the implementation of accessibility features

Although there are many arguments for creating accessible websites, some designers still argue that it is untenable to expect accessibility to be a top priority of designers. Common arguments include the following:

1. Creating accessible websites is too expensive;
2. Creating accessible websites is too difficult or time-consuming;
3. Accessible web design is at odds with good graphic design; and
4. Accessibility features will only benefit a small portion of users.

In the following sections, I will address each of these arguments and provide evidence that accessible web design is not antithetical to good and inexpensive web design.

Creating accessible websites is too expensive. Although the cost of designing and maintaining a website is still relatively expensive in terms of upfront investment, it is now cheaper to host websites and easier in terms of server connection configuration. Also, although websites can be expensive, they generate large amounts of revenue for companies, creating trillions a year in profits for global companies like Amazon, Alibaba, Google, and Facebook. In “The business case for digital accessibility,” the W3C (2018e) also points out that “Businesses that integrate accessibility are more likely to be innovative, inclusive enterprises that reach more people with positive brand messaging that meets emerging global legal requirements” (“Is there a business case for accessibility”).

Further, the W3C argues that accessibility often drives innovation because accessible design features create “options that are useful for people with and without disabilities” (“Drive innovation”). Companies that take accessibility issues seriously, such as Apple, also gain more customer loyalty and therefore increase their market reach (“Increase market reach”). Finally, the W3C argues that making websites accessible minimizes legal risks, especially as more countries are adopting accessibility requirements for technology (“Minimize legal risk”). In fact, the number of ADA lawsuits against companies with inaccessible websites is expected to top 2,000 by the end of 2018, which represents a 90% increase in web accessibility lawsuits from 2017 (Martin 2018). The most recent accessibility lawsuits have been filed against several

major organizations, including Winn-Dixie, Domino's Pizza, Harvard, MIT, Kmart, Hulu, and CVS (Martin 2018). Although websites can be somewhat expensive to design and maintain, creating accessible websites can drive innovation, improve market reach, and help organizations avoid costly ADA lawsuits.

Creating accessible websites is too difficult or time-consuming. As I have demonstrated in my redesign of the LinkedIn mobile and Wikipedia web pages (See Appendices), implementing a few basic accessibility features is neither excessively difficult nor particularly time-consuming, especially for existing websites. Further, the creation of standardized web design languages and robust tools makes it easier than ever to create websites. Most web designers do not code pages by hand, and plenty of prominent organizations use tools like WordPress to design websites.

Accessible web design is at odds with good graphic design. The point of accessibility is not to make websites aesthetically inferior, but to make them simpler to use. Good web design and accessible design share the same minimalist principles that benefit all users. For example, a color switcher only modifies the look of a page upon a user's request, and some users without disabilities prefer darker styles over light ones. Concerning alt text and input labeling, neither of these practices has a significant impact on the aesthetic value of web pages. Alt text is not perceivable to sighted users unless images do not load – in which case alt text benefits them as well – and input labels have little or no impact on the layout of forms. In my LinkedIn mobile web page example, my modifications didn't alter the look of the page at all because I used [aria](#) attributes, which, like alt text, are not perceptible to sighted users. Although traditional input

labeling might be less aesthetically attractive than relying on placeholder text, having clearly labeled input fields helps both users with disabilities and users without disabilities.

Accessibility features will only benefit a small portion of users. Accessible designs have been proven to benefit most users. For example, the adoption of sans-serif fonts in electronic mediums is not only easier to read for people with dyslexia, but also for most users. The U.S. Census (2017) estimates that 26.7 million Americans live with disabilities that significantly impact their lives (“Population distribution”). Further, more than half a billion people worldwide have significant visual impairments, and that number is expected to grow as the postwar generations continue to age (WHO 2018). The CDC (2009) estimates that nearly 30 million people in the U.S. live with age-related visual impairments and that that number is expected to double by 2020 as more of the population continues to age and the prevalence of chronic diseases like diabetes continues to rise (“Estimated growth in population”). Moreover, people with reading and learning disabilities, such as dyslexia, often benefit from the use of screen readers and websites with low luminance theme options. Even by only implementing the features discussed in this research, web designers have the potential to make the web more accessible to billions globally.

CONCLUSION

Although technical writers and web designers may have a challenge of implementing all accessibility features in their website designs, they have the potential to make the web more accessible to large portions of their audience, especially as more of the world continues to gain Internet access. As Jakob Nielsen (2000) puts it, “Even if you cannot design a fully-accessible site, you have the responsibility to include as many accessibility features as possible. Many are, in fact, quite easy and cheap” (p. 311). Technical writers have both the obligation and the ability to make web accessibility a top issue in web design. Technical writers have the platform to ensure that accessibility principles become industry best practices on par with other usability issues like writing well and content design.

While this research is fairly limited in its scope and perspective, it provides a reasonable approach for technical writers and other web designers to make their websites more accessible and advocate for the widespread adoption of accessible web design practices. Future research could include other aspects of visual accessibility, or areas such as accessibility for people with auditory, speech, motor, and cognitive impairments, and accessibility for mobile devices or new technologies. My hope is that this research, along with any further research that it may inspire, helps technical writers integrate accessibility principles into their designs and encourages the use of heuristics-based website usability research among technical writers who design websites.

OTHER RESOURCES

The following list of resources includes links that might be helpful in creating more accessible websites, such as design standards and guidelines, advocating for users with disabilities and tools that aid designers in creating and testing website accessibility.

General web design standards and guidelines

W3C resources on web accessibility

<https://www.w3.org/standards/webdesign/accessibility>

Department of Health and Human Services usability guidelines

<https://www.usability.gov/accessibility>

Mobile web design guidelines

Mozilla developer's network mobile accessibility checklist

https://developer.mozilla.org/en-US/docs/Web/Accessibility/Mobile_accessibility_checklist

BBC's mobile accessibility principles

<https://www.bbc.co.uk/guidelines/futuremedia/accessibility/mobile/principles>

Material Design guidelines for mobile and web user interfaces

<https://material.io/design/>

Advocacy

How to advocate for users with disabilities by reporting inaccessible websites

<https://www.w3.org/WAI/teach-advocate/contact-inaccessible-websites/>

Tools

Icons for web and mobile app design

<https://material.io/tools/icons/>

Free, open-source Non-Visual Desktop Access (NVDA) screen reader

<https://www.nvaccess.org/>

REFERENCES

- Anderson, M. and A. Perrin (2017). Disabled Americans are less likely to use technology. Retrieved from <http://www.pewresearch.org/fact-tank/2017/04/07/disabled-americans-are-less-likely-to-use-technology/>
- Carey, M. and M. McFadden Lanyi, and D. Longo, and E. Radzinski, and S. Rouiller, and E. Wilde. (2014). *Developing quality technical information: A handbook for writers and editors*. Third Edition. IBM Press.
- CDC (2009). The burden of vision loss. Retrieved from https://www.cdc.gov/visionhealth/basic_information/vision_loss_burden.htm
- Chowdhury, G. G. & Chowdhury, S. (2011). *Information users and usability in the digital age*. London, England: Facet.
- Lazar, J., Allen, A., Kleinman, J., & Malarkey, C. (2007). What frustrates screen reader users on the web: a study of 100 blind users. *International Journal of Human-Computer Interaction*, 22(3), 247–269. doi: 10.1080/10447310701373063
- Martin, H. (2018). Lawsuits targeting business websites over ADA violations are on the rise. *LA Times*. Retrieved from <http://www.latimes.com/business/la-fi-hotels-ada-compliance-20181111-story.html>
- Nielsen, J. (2000). *Designing web usability*. Berkeley, CA: New Riders.
- Nielsen, J. (1994). Ten usability heuristics for user interface design. Enhancing the explanatory power of usability heuristics. Proc. ACM CHI'94 Conf. (Boston, MA, April 24-28), 152-158. Retrieved from <https://www.nngroup.com/articles/ten-usability-heuristics/>

Redish, J. (2012). *Letting go of the words: writing web content that works*. Waltham, MA: Morgan Kaufmann.

Steen-Hansen L. and Siri Fagernes (2016). The importance of process-oriented accessibility guidelines for web developers. In *Universal design 2016: learning from the past, designing for the future: Proceedings of the 3rd International Conference on Universal Design (UD 2016), York, UK, August 21-24, 2016*. Amsterdam, Netherlands: IOS Press.

U.S. Census Bureau (2017). Facts for features: Anniversary of Americans with Disabilities Act: July 26. Retrieved from <https://www.census.gov/newsroom/facts-for-features/2017/cb17-ff11-disabilities.html>

U.S. Dept. of Health and Human Services (2006). *Research-based web design and usability guidelines*. Retrieved from https://www.usability.gov/sites/default/files/documents/guidelines_book.pdf

U.S. General Services Administration (2017). Federal social media accessibility toolkit hackpad. Retrieved from <https://digital.gov/resources/federal-social-media-accessibility-toolkit-hackpad/>

W3C (2016). G17: Ensuring that a contrast ratio of at least 7:1 exists between text (and images of text) and background behind the text. Retrieved from <https://www.w3.org/TR/WCAG20-TECHS/G17.html>

W3C (2017). Easy checks – A first review of web accessibility. Retrieved from <https://www.w3.org/WAI/test-evaluate/preliminary/>

W3C (2018a). Tips for getting started designing for web accessibility. Retrieved from

<https://www.w3.org/WAI/tips/designing/>

W3C (2018b). How to meet WCAG 2 (Quick reference). Retrieved from

<https://www.w3.org/WAI/WCAG21/quickref/>

W3C (2018c). Introduction to web accessibility. Retrieved from

<https://www.w3.org/WAI/fundamentals/accessibility-intro/>

W3C (2018d). Involving users in evaluating web accessibility. Retrieved from

<https://www.w3.org/WAI/test-evaluate/involving-users/>

W3C (2018e). The business case for digital accessibility. Retrieved from

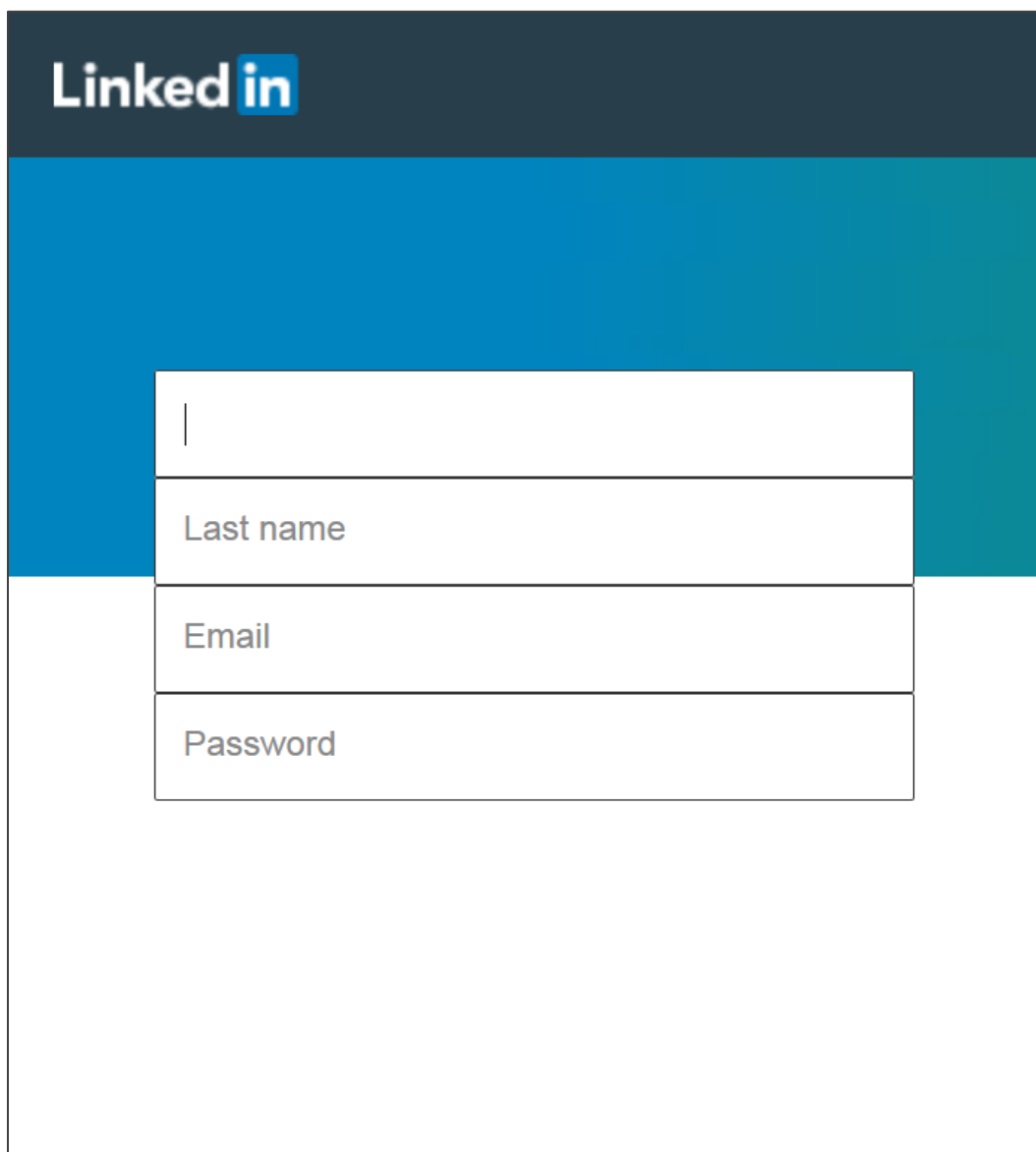
<https://www.w3.org/WAI/business-case/>

WHO (2018). Blindness and vision impairment. Retrieved from

<http://www.who.int/news-room/fact-sheets/detail/blindness-and-visual-impairment>

APPENDICES

Appendix A: [LinkedIn mobile mockup before](#)



The image shows a mobile login page for LinkedIn. At the top, there is a dark blue header with the LinkedIn logo. Below the header is a large blue gradient area. In the center, there is a white login form with four input fields: a first name field (with a vertical line cursor), a last name field, an email field, and a password field. The form is set against a white background that is part of the overall page layout.

Figure 2: LinkedIn mobile login page before changes.

Appendix B: [LinkedIn mobile mockup after](#)

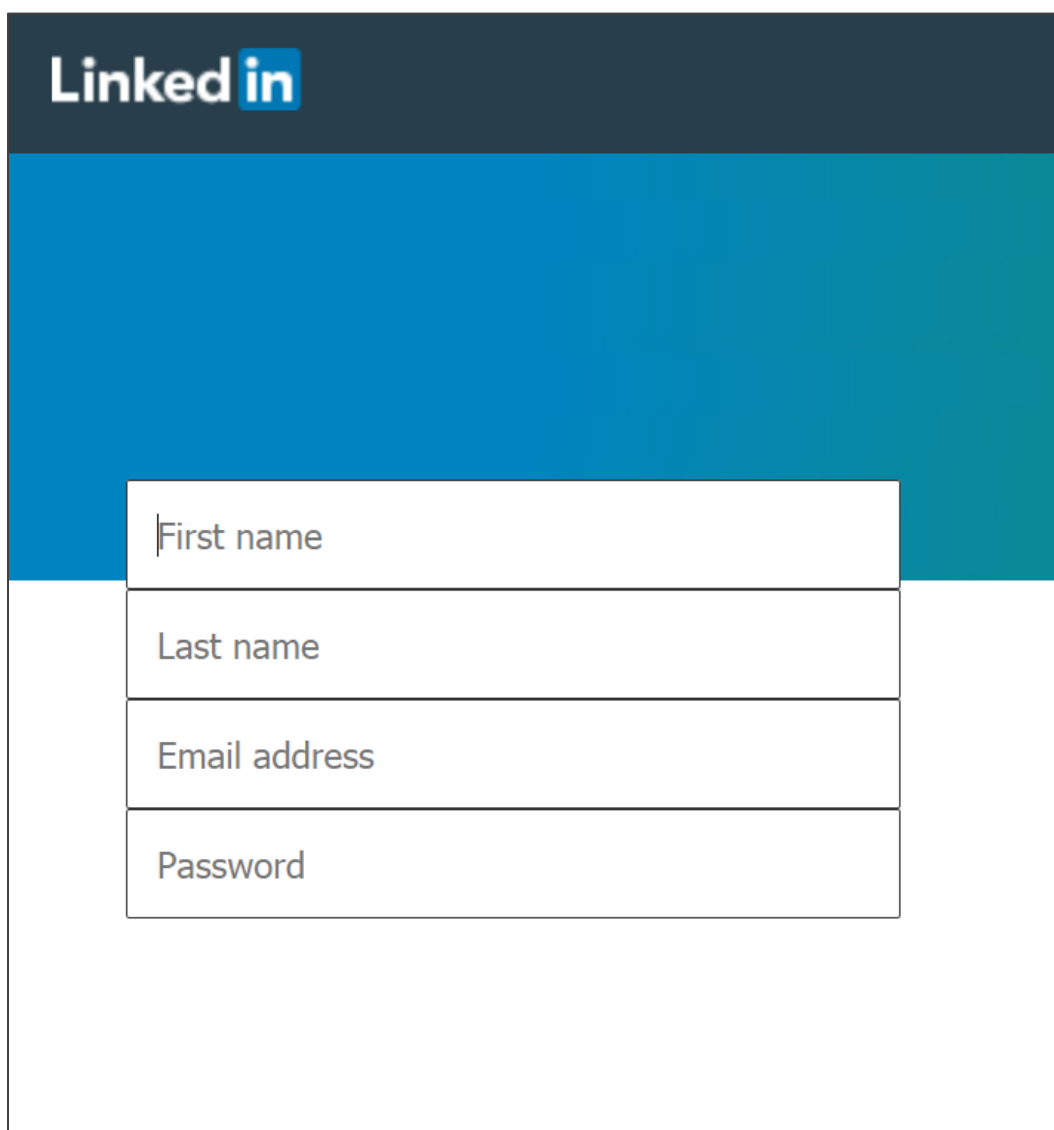
A mockup of the LinkedIn mobile login page. The top section has a dark blue header with the LinkedIn logo. Below this is a large blue gradient area. At the bottom, there is a white login form with four input fields: 'First name', 'Last name', 'Email address', and 'Password'. The form is centered on the page.

Figure 3: LinkedIn mobile login page after changes

Appendix C: LinkedIn form source code before

LinkedIn mobile Before

This mockup is LinkedIn's mobile login page. Errors in markup, along with an unconventional use of JavaScript to make disappearing labels instead of using placeholders make this site inaccessible to screen readers.

The autofocus in the original source code was also on the email field, which might confuse someone who cannot see the screen and/or naturally assume that the cursor was at the top of the form.

NOTE: The following code sample contains syntax highlighting commonly used text editors to make code easier to read. Depending on the language, bolded and colored items might include: Functions, classes, attributes, values, statements (like **if**, **else**, **for**, or **while**), logical operators (like **AND**, **OR**, **TRUE**, and **FALSE**), and so on. Similarly, the ellipses (...) signify that code is irrelevant to the example and it has been omitted for the sake of brevity.

```
...
<script>

    function deltext(val){
        val.style.display='none'
    }

    function replacetext(val, vn){
        if(vn == ''){
            val.style.display='block'
        }
    }
</script>

...
<div class="signin">
<form id="testform" method="post" action="#">
    <section class="form-orig">

        <label for="login-firstname" class="origlabel" id="firstname">First
            name</label>
        <input
            type="text" class="lgtextbox" name="fn" autocomplete="off"
            placeholder="" aria-required="true"
            onfocus="deltext(firstname)"
            onblur="replacetext(firstname, this.value)"/><br>

        <label for="login-lastname" class="origlabel" id="lastname">Last
            name</label>
        <input
            type="text" class="lgtextbox" name="ln" autocomplete="off"
            placeholder="" aria-required="true" onfocus="deltext(lastname)"
```

```

        onblur="replacetext(lastname, this.value)"/><br>

<label for="login-email" class="origlabel" id="email">Email</label>
<input
    type="text" class="lgtextbox" name="em" autocomplete="off"
    placeholder="aria-required="true" onfocus="deltext(email) "
    onblur="replacetext(email, this.value) "
    autofocus="autofocus"/><br>

<label for="login-pw" class="origlabel"
    id="password">Password</label>
<input
    type="text" class="lgtextbox" name="pw" autocomplete="off"
    placeholder="aria-required="true" onfocus="deltext(password) "
    onblur="replacetext(password, this.value)"/><br>
...

```

Appendix D: LinkedIn form source code after

LinkedIn mobile After

This section is the repaired version of the LinkedIn mobile login site. I have removed the JavaScript and labels altogether and replaced them with placeholders for sighted users and aria-label for people who are using screen readers. I have also changed the focus to the first input in the form (first name) to avoid confusing users who are using screen readers.

NOTE: The following code sample contains syntax highlighting commonly used with text editors to make code easier to read. Depending on the language, bolded and colored items might include: Functions, classes, attributes, values, statements (like **if**, **else**, **for**, or **while**), logical operators (like **AND**, **OR**, **TRUE**, and **FALSE**), and so on. Similarly, the ellipses (...) signifies that code is irrelevant to the example and it has been omitted for the sake of brevity.

```

...

<div class="signin">
    <form method="post" action="#">

```

```

<label class="col-label" id="firstname"></label>
  <input type="text" class="lgtextbox" name="fn"
    autocomplete="off" aria-label="First name" aria-
    required="true" placeholder="First name" autofocus><br>

<label class="col-label" id="lastname"></label>
  <input type="text" class="lgtextbox" name="ln"
    autocomplete="off" aria-label="Last name" aria-
    required="true" placeholder="Last name"><br>

<label class="col-label" id="emailaddress"></label>
  <input type="text" class="lgtextbox" name="email"
    autocomplete="off" aria-label="Email address" aria-
    required="true" placeholder="Email address"><br>

<label class="col-label" id="password"></label>
  <input type="text" class="lgtextbox" name="pw"
    autocomplete="off" aria-label="Password" aria-
    required="true" placeholder="Password"><br>

</form>
</div>
...

```

Appendix E: Wikipedia mockup before

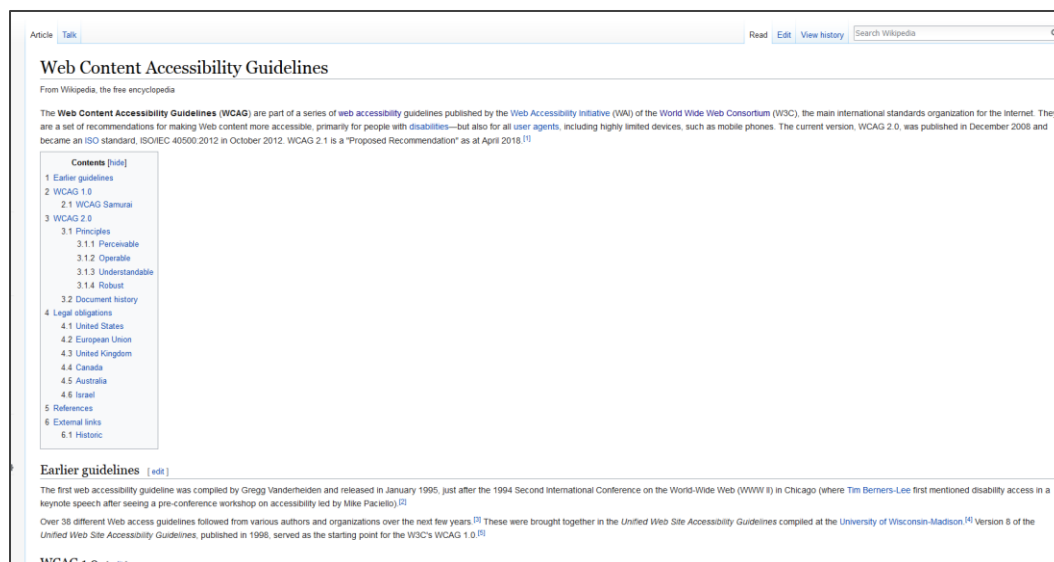


Figure 4: Wikipedia page mockup before changes

Appendix F: Wikipedia mockup after – high luminance (standard theme)

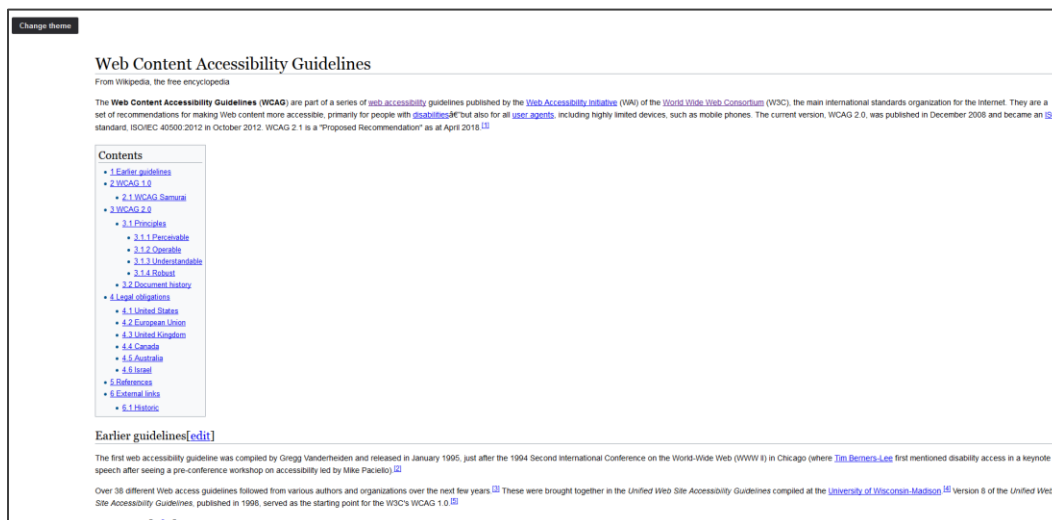


Figure 5: Wikipedia page mockup after changes with high luminance theme

Appendix G: Wikipedia mockup after – low luminance (dark theme)

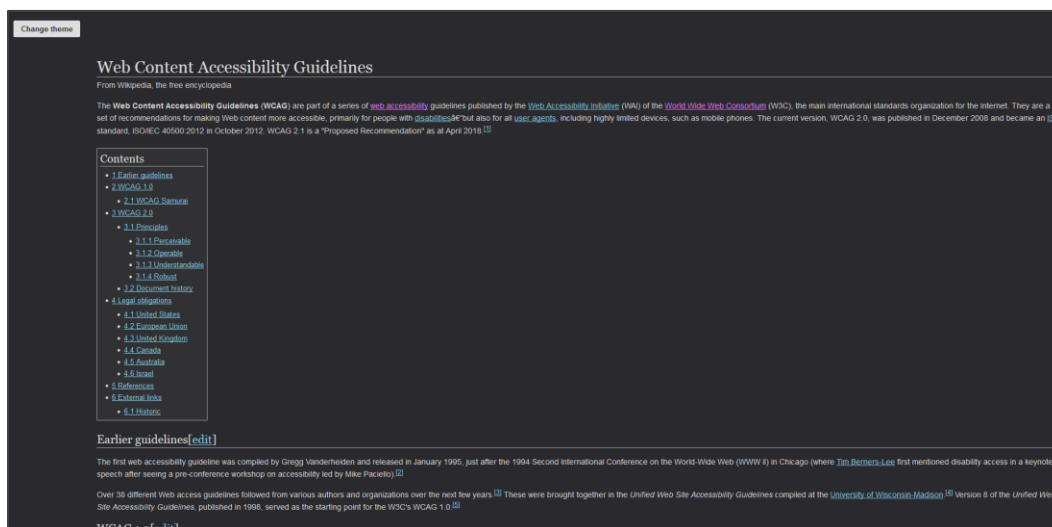


Figure 6: Wikipedia page mockup after changes with low luminance theme.

Appendix H: Wikipedia HTML and JavaScript changes

Wikipedia After

To implement the low luminance theme switcher, I added a button, a line of JavaScript, and a few lines of CSS. When the user clicks the button, the JavaScript

toggles a class for the page's body. That class is then tied to CSS modifications, such as background and text color. The rest of the page is merely a copy of the original.

NOTE: The following code sample contains syntax highlighting commonly used with text editors to make code easier to read. Depending on the language, bolded and colored items might include: Functions, classes, attributes, values, statements (like **if**, **else**, **for**, or **while**), logical operators (like **AND**, **OR**, **TRUE**, and **FALSE**), and so on. Similarly, the ellipses (...) signifies that code is irrelevant to the example and it has been omitted for the sake of brevity.

```
...
    <button id="change-theme">Change theme</button>
    <script>
        document.getElementById("change-theme").addEventListener("click",
function() {
            document.body.classList.toggle("dark-theme");
        });
    </script>
...
```

Appendix I: Wikipedia CSS changes

CSS changes for low luminance theme. Most of these changes were to existing classes like the TOC and required little modification.

NOTE: The following code sample contains syntax highlighting commonly used text editors to make code easier to read. Depending on the language, bolded and colored items might include: Functions, classes, attributes, values, statements (like **if**, **else**, **for**, or **while**), logical operators (like **AND**, **OR**, **TRUE**, and **FALSE**), and so on. Similarly, the ellipses (...) signifies that code is irrelevant to the example and it has been omitted for the sake of brevity.

Styles for the button

```
#change-theme{
  font-family: sans-serif;
  border: 0px;
  outline: 0px;
  height: 30px;
  width: 120px;
  border-radius: 2px;
  background-color: #2a2a2e;
  color: #eee;
  margin-top: 10px;
  font-weight: bold
}
```

Makes the switch between themes less jarring by slowing the transition between light and dark by 0.1 seconds and easing the transition in.

```
body{
  transition: all 0.1s ease-in;
}
```

Primary background/text inversion. This is a class that is added to the page's body on toggle that inverts the background and text colors.

```
.dark-theme{
  background-color: #2a2a2e;
  color: #ddd;
}
```

Changes the colors of headings 1-4 from dark to light.

```
.dark-theme h1, .dark-theme h2, .dark-theme h3, .dark-theme h4{
  color: #ddd;
}
```

Changes the color of links from standard blue to a lighter shade of blue to provide more contrast on the darker background.

```
.dark-theme a{
  color: skyblue;
}
```

Changes the color of visited links from standard purple to a lighter shade of purple to provide more contrast on the darker background.


```
.dark-theme a:visited{
  color: #eb7fff;
}
```

Inverts the button colors for each theme so it remains highly contrasted and therefore visible to the user.

```
.dark-theme #change-theme{
  background-color: #ddd;
  color: #2a2a2e;
  font-weight: bold;
}
```

Inverts the TOC colors to match the darker theme.

```
.dark-theme .toc, .dark-theme .mw-warning, .dark-theme
.toccolours {
  border: 1px solid #aaa;
  background-color: #313133;
}
```

Inverts the color of bullets (unordered list items) to provide more contrast with the darker theme.

```
.dark-theme ul{
  list-style: disc;
  list-style-image: url("images\light-bullet.svg");
}
```