Accessible CSS: Proposal for an Accessible, Comprehensive Validation Tool for Screen Reader Users

Claire Kearney-Volpe
New York University
New York, NY, USA
claire.kv@nyu.edu

Haoran Wen
Rutgers University
New Brunswick, NJ, USA
haoranwen1@gmail.com

Amy Hurst
New York University
New York, NY, USA
amyhurst@nyu.edu

ABSTRACT
As the Internet and web-enabled technologies become ubiquitous and there is greater need for web-related jobs, there is a lack of diversity and representation by persons with disabilities. One factor contributing to this is that the production ——of web technologies presents various accessibility barriers for individuals that are blind or low vision. CSS and visual styling are areas of particular stumbling blocks that lacks easy, accessible, and comprehensive tools for nonvisual CSS validation. CSS is a core language and component of the web used to describe visual representations and due to the visual nature of CSS nonvisual developers struggle to with its use: often time relying on sighted third party member to assist with validating their CSS. In order to striving for a more diverse participation, better accessibility support, and greater independence of blind or low vision web developers we evaluated existing CSS tool to aim for creating a accessible CSS validation tool that would allow blind and low vision web developers to build, test, and produce websites and web applications with greater confidence and independence.

Author Keywords
Accessibility; Web Development; CS Education; Design; Human-Computer Interaction

INTRODUCTION
As the Internet and web-enabled technologies become ubiquitous and there is greater need for web-related jobs. However, there is a lack of diversity and representation by persons with disabilities in the US workforce. Although employment in Computing and Information Systems reached close to 4 million as of 2016 with significant growth predicted for Web Development and Web Development adjacent jobs [6], persons with disabilities are significantly underrepresented [19].

This underlying reasons for underrepresentation is complex, and includes socio-economic, educational, and systemic barriers, but also a lack of accessibility in the tools for design and development. For screen reader users, significant barriers include the lack of accessibility informational resources, inaccessible editors, and coding environments [3, 18]. With regard to web interface design and visual styling, this task is extremely difficult for people that are blind. We describe our work investigating the opportunity and need for accessible CSS validation tools to increase participation in web-development and adjacent fields.

RELATED WORK

Programming and Design Accessibility
Notable past work to increase computing diversity explored making programming languages, software, and curricula more accessible to blind students. This work has largely been in the development of auditory interfaces, development environment plug-ins for navigation, and novel programming languages [4, 23, 24]. The user interface design process is integral with web-development, and unfortunately, little research has focused on accessibility. Norman et al. found that nearly all of their blind participants created websites collaboratively or kept CSS stylesheets created by sighted developers on-hand as a reference [20]. Bennet et al. [5] conducted workshops in which participants used a variety of common craft supplies for their tactile qualities, improving design process accessibility (specifically the ideation process).

CSS Validation
CSS is a core language of the web used to describe the visual presentation of sites [17]. The process of validating CSS starts with parsing CSS source code into an abstract syntax tree structure. Once parsed into an abstract syntax tree, validators traverse the tree comparing and verifying the syntax against a chosen schema.

METHODS AND FINDINGS
We surveyed existing CSS support software by searching Github [12], Google Scholar [13], and the ACM Digital Library [2] using the search terms: CSS Tool, Design Tool, CSS Validator, and Accessibility Checker. Through this process, we found 10 CSS validation tools and evaluated them for type, install/set-up, and accessibility.

We categorized each tool into one of three types: 1) CSS Syntax, 2) Computed Style, and finally 3) Automated Accessibility validators that in analyze semantic html, and evaluate color contrast, text size, element spacing, and other visual properties. Our second categorization, install and setup, refers to the installation process for this tool. These tools worked through the command line, through a web page, or browser extension. Finally, our "accessibility category relates to use with a screen reader. We tested each validator with Voiceover using the Safari web browser. For syntax and automated accessibility validators that operate through the command line, web app, or browser extension, each had good support for use with a screen reader. Conversely, the Computed Style tools had variable support for use with a screen reader. Each of the Computed Style tools that we evaluated operated through the command line or by including a reference to a library in the user's code. This install/set up is accessible, however the outputs of these tools had variable support for use with a screen reader. For example, the Computed Variable [11] tool is a JavaScript plugin and results are rendered as visual indicators (drop shadows, etc.) on elements and only on hover by default. Table 1 shows examples of each type of validator with information about install/set up and accessibility.

We also used this survey to better understand how each type works. CSS Syntax validators compare written CSS to the W3C CSS standards. Computed Value validators extract rendered CSS from browsers and compare these values with written CSS, or a design schema. And finally, Automated Accessibility validators are able to parse extrapolate and check for color values, tab index, properly label properties, and aria tags and other defined properties.

Name	Validator Type	Install/Set up	Accessibility Support
Jigsaw	CSS Syntax	Web App, with style sheet uploader	Yes
Computed Style to in-line Style	Computed Style	Command line tool requiring NPM install	No
Wave	Automated Accessibility	Web App, and browser extension	Yes

Table 1. Survey and evaluation of CSS validation tools.

Each of these tools have use in a developer workflow based on type, however none are comprehensive (incorporate each type of CSS validation), and there is inconsistent support for beginner web developers and screen reader accessibility.

Gaps and Opportunities For Teaching and Learning

Our past work teaching web development to screen reader users confirms the need for a simplified, comprehensive, and accessible software tool, and the impact of not having one.

During a 7-day Web-Development training, we taught adult screen reader users HTML, CSS, and JavaScript at a technology camp in Uganda [15]. During a unit on CSS, we used a color naming tool that we previously created with mixed success. This web-based tool took red, green, and blue values from a text input, generated a web color and color name, and announced the name to screen readers. Students that had partial sight or had lost their vision were much more responsive to the tool. Students that experimented with using both foreground and background colors, had difficulty selecting pleasing color contrasts for sighted users was a challenge (Figure 1). In a post-course focus group, participants expressed wanting more support and would prefer to use a design framework/system that helped with visual design (specifically grid layouts and color palettes).

In a 10-day intensive Web-Development workshop at New York Public Library's Braille and Talking Book Library, we observed 14 beginner participants struggle with using command line tools, incorporating box-model/layout styles, and sourcing and using images without distortion/pixilation. At the time of this workshop (August 2018), our participants struggled because Webaim's color contrast analyzer [28] did not support use with a screen reader.

Figure 1. Blind student's website with color contrast issues of blue text on a green background with a red border.

Drawing from all of our observations teaching web development to screen reader users, we have noted the following use cases for a CSS tool: 1. CSS syntax is correct, but font files not loaded/referenced properly, 2. An image file is loaded, but rendered too large and becomes distorted/skewed/pixelated, 3. CSS is written properly, but color contrast fails WCAG standards, and 4. Padding and margins are rendered as text in CSS, but doesn't match with the rest of the page's content.

RESULTS

Based on our literature review, validator evaluation, and observations of screen reader users working with CSS, we have identified the need for a new comprehensive, accessible CSS validation tool. The current tools we found do one or two things really well, but do not satisfy all of the use cases identified. So we developed a CSS validator tool that verifies CSS syntax, Computed Styles (checking against design system schemas), and Accessibility validation. The validator functions as a website where the user would upload their code for validation and the respective results would be displayed to them. This tool is developed with the target audience of screen readers users who are brand new to development in mind.

CONCLUSION

Ubiquity of web and web-enabled technologies presents greater risk of accessibility-related barriers to participation in the digital economy. We are also faced with opportunities to create a more diverse and accessibility-aware workforce. With this work, we contribute an overview of existing Web-Development Style Tools (CSS validators), and use-cases

based on our observations of screen reader users implementing CSS. Finally, we propose the design of a new comprehensive, easy-to-use, and accessible CSS Validator.

ACKNOWLEDGMENTS

We thank our research/workshop participants. We also thank the NYU ability project, Access Computing, and the CRA-W DREU program for their support.

REFERENCES

1. Accessibility Insights for Web. Retrieved from https://accessibilityinsights.io/docs/en/web/overview
2. ACM Digital Library. Retrieved from https://dl.acm.org/
3. K. Albusays, and S. Ludi. 2016. Eliciting programming challenges for developers with visual impairments survey. Retrieved December 12, 2019 from https://people.rit.edu/ kla3145/Research/pdf/BlindProgrSurvey.pdf
4. Catherine Baker, Lauren Milne, and Richard Ladner. 2015. Structjumper: A tool to help blind programmers navigate and understand the structure of code. In Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems, 3043–3052. DOI: https://.org/citation.cfm?id=2702589
5. Cynthia L. Bennett, Kristen Shinohara, Brianna Blaser, Andrew Davidson, and Kat M. Steele. 2016. Using a Design Workshop To Explore Accessible Ideation. In Proceedings of the 18th International ACM SIGACCESS Conference on Computers and Accessibility (ASSETS '16). ACM, New York, NY, USA, 303-304. DOI: https://doi.org/10.1145/2982142.2982209
6. Bureau of Labor Statistics. Occupational Outlook Handbook: Web Developers. Retrieved from https://www.bls.gov/ooh/computer-and-information-technology/web-developers.htm#tab-8
7. Computed Style to in-line Style. Retrieved from https://github.com/lukehorvat/computed-style-to-inline-style
8. Crass. Retrieved from https://github.com/rgrove/crass/
9. CSS Accessibility Validator. Retrieved from https://github.com/elad2412/css-accessibility-validator
10. CSS-Check. Retrieved from https://github.com/elifesciences/css-check
11. Computed-Variables. Retrieved from: https://github.com/tomhodgins/computed-variables
12. Github. Retrieved from https://github.com/
13. Google Scholar. Retrieved from http://scholar.google.com/
14. JQuery computed-style plugin. Retrieved from https://github.com/jamierumbelow/jquery.computed-style
15. Claire Kearney-Volpe, Scott Fitzgerald, and Amy Hurst. 2019. Blind Web Development Training at Oysters and Pearls Technology Camp in Uganda Proceeding from the 16th International Web for All Conference: Addressing Information Barriers, San Francisco, CA, USA. (to appear)
16. Ali Mesbah and Shabnam Mirshokraie. 2012. Automated analysis of CSS rules to support style maintenance. In Proceedings of the 34th International Conference on Software Engineering (ICSE '12). IEEE Press, Piscataway, NJ, USA, 408-418.
17. Mozilla. CSS: Cascading Style Sheets. Retrieved from https://developer.mozilla.org/en-US/docs/Web/CSS
18. Rahul Kumar Namdev and Pattie Maes. 2015. An interactive and intuitive stem accessibility system for the blind and visually impaired. In Proceedings of the 8th ACM International Conference on Pervasive Technologies Related to Assistive Environments (PETRA '15). ACM, New York, NY, USA, Article 20, 7 pages. DOI: https://doi.org/10.1145/2769493.2769502
19. National Science Foundation. 2017. Women Minorities, and Persons with Disabilities in Science and Engineering. Retrieved from https://www.nsf.gov/statistics/2017/nsf17310/static/downloads/nsf17310-digest.pdf
20. Kirk Norman, Yevgeniy Arber, and Ravi Kuber. 2013. How accessible is the process of web interface design?. In Proceedings of the 15th International ACM SIGACCESS Conference on Computers and Accessibility (ASSETS '13). ACM, New York, NY, USA, , Article 51 , 2 pages. DOI: http://dx.doi.org/10.1145/2513383.2513385
21. PrettyCSS. Retrieved from https://github.com/fidian/PrettyCSS
22. Henrik Svarrer Larsen and Per-Olof Hedvall. 2012. Ideation and ability: when actions speak louder than words. In Proceedings of the 12th Participatory Design Conference: Exploratory Papers, Workshop Descriptions, Industry Cases - Volume 2 (PDC '12), Vol. 2. ACM, New York, NY, USA, 37-40. DOI=http://dx.doi.org/10.1145/2348144.2348157
23. J. Sánchez, and F. Aguayo. 2004. “Listen what I do: Blind Learners Programming Through Audio,” Memorias TISE, 120-124.
24. A. Stefik, C. Hundhausen, and D. Smith. 2011. On the design of an educational infrastructure for the blind and visually impaired in computer science. In Proceedings of the 42nd ACM technical symposium on Computer science education, ACM, (March 2011), 571-576.
25. Wave (Web-based). Retrieved from https://wave.webaim.org/

26. Wave (browser plug-in). Retrieved from https://wave.webaim.org/extension/
27. W3C Jigsaw. Retrieved from https://jigsaw.w3.org/css-validator/
28. Webaim. Web Accessibility in Mind: Color Contrast Checker. Retrieved from https://webaim.org/resources/contrastchecker/