

WebTrax: Visualizing Non-Visual Web Interactions

Jeffrey P. Bigham and Kyle Murray

Department of Computer Science
University of Rochester
Rochester, NY 14627
jbigham@cs.rochester.edu, kyle.murray@rochester.edu

Abstract. Web accessibility and usability problems can make evaluation difficult for non-experts who may be unfamiliar with assistive technology. Developers often (i) lack easy access to the diversity of assistive technology employed by users, and (ii) are unaware of the different access patterns and browsing strategies that people familiar with a specific assistive technology tool might use. One way to overcome this problem is to observe a person with a disability using their tools to access content, but this can often be confusing because developers are not familiar with assistive technology and tools are not built supporting this use. In this paper we introduce *WebTrax*, a tool that we have developed to support developers who engage blind web users as part of their accessibility evaluation or education strategy. WebTrax helps visualize the process that screen reader users employ to access content, helping to make problems more obvious and understandable to developers.

Key words: web accessibility, visualization, screen reader, web trails

1 Introduction

We introduce *WebTrax*, a tool that visualizes the non-visual web browsing of screen reader users to help sighted developers better understand how blind people access web content with a screen reader.

The diverse issues involved in creating accessible web content can make development a frustrating, subjective, and difficult task. Guidelines and standards serve as a valuable starting point for developers hoping to create accessible web content, but many subjective and practical factors influence realized accessibility. Accessibility and usability are often linked [11], and if a resource is not usable then it is in effect not accessible even if it is technically accessible.

In this paper, we explore a different, complementary approach that enables web developers to gain a better understanding of the usability of their own sites. Screen readers are complex software packages that can be intimidating and difficult to understand for those without adequate experience. Even watching a skilled user “drive” a screen reader can be confusing - the screen reader speaks a wide array of content, some derived from the web page it is reading and

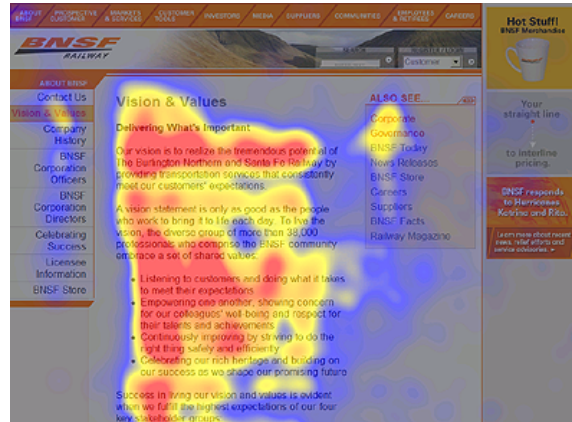


Fig. 1. In this visualization from [useit.com](#), a user’s attention has been captured by an eye tracker and visually represented as a heatmap over the original page. WebTrax provides analogous visualizations of non-visual attention and usability.

some meant as helpful information for the user. Something as simple as “link ‘Search’” or “Table 4, 3 rows, 2 columns” can be confusing to someone who has not heard it before. Most existing screen readers either do not provide support for highlighting what is being read or provide only minimal support. To understand usability, it would be useful to have a non-visual equivalent of the heatmaps generated by eye-tracking data (Figure 1). WebTrax visualizes the content that is being read in order to help sighted observers make sense of the screen reader experience.

WebTrax is built into the web-based screen reader WebAnywhere [3]. WebAnywhere is unique among screen readers in its ability to be run anywhere - on any platform, from any browser, without requiring special permissions. This means that WebTrax can also be run from any computer, which is an important advantage for a tool that we hope will support learning about accessibility. Educators or WebAnywhere users can demonstrate the non-visual browsing experience to people wherever they happen to be without extensive setup.

1.1 Summary of Contributions

This work makes the following contributions:

- We motivate visualizing non-visual web interactions to help developers improve their understanding of how screen reader users browse through content.
- We describe three visualizations for non-visual web browsing: basic, phosphorous, and heatmaps.
- We present implementations of these visualizations that are integrated into the WebAnywhere non-visual web browser so they are accessible anywhere.

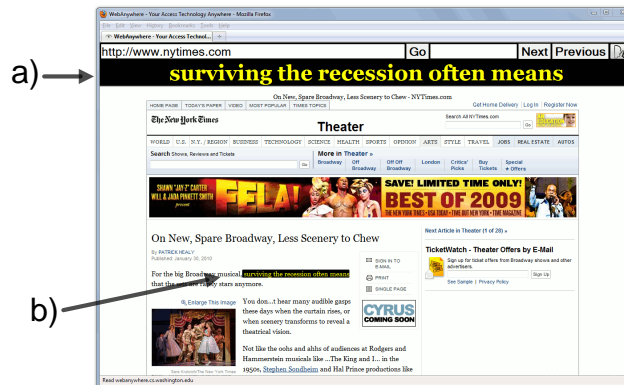


Fig. 2. The basic visualization. (a) The text currently being read is shown above the content in a high-contrast, magnified view to assist those reading along and those with low-vision. (b) The current content being read is highlighted so that users and developers can more easily read along with WebAnywhere.

2 Related Work

WebTrax is designed as both an interactive tool to help web developers better understand the browsing they are observing screen reader users perform and as an educational tool to help teachers convey subtle issues concerning web accessibility and usability.

Accessibility Evaluators: Many accessibility validators have been created to help developers evaluate and improve their web pages. Popular tools include Bobby [17], FAE [5], and WAVE [18]. Developers point these tools at their web pages, and receive back a list of potential accessibility problems. Although a valuable first step in ensuring that one's web page is accessible, evaluators have two primary shortcomings. First, validators cannot evaluate accessibility issues that cannot be detected automatically; for example, evaluation tools can detect if an image lacks alternative text, but they cannot judge if that alternative text is appropriate and informative [4]. Second, usability problems, such as problems with reading order or lack of heading tags or other markup, are difficult to detect automatically. Such usability problems are likely to become more pervasive as web pages become more complex and begin to behave more like applications than static documents [12].

As a fallback, evaluation tools present warnings for content that may have a problem. Evaluation tools often present so many warnings that novice developers might be overwhelmed or be discouraged from fixing anything. While those who are motivated will investigate all warnings, typical developers may be unwilling to investigate each issue, especially when many (or most) will end up not being problems. As an example, WAVE displays a warning anytime a `tabindex` attribute is used within a web page. This attribute can be used correctly to enforce a meaningful tab order, but when used incorrectly results in a confusing

ordering. WebTrax can help visualize how real users might browse this content and expose whether the content is truly a concern.

Simulation: Another approach to evaluation is to provide views of content that simulate what a disabled user might experience. For instance, ADesigner can visually simulate the usability of a web site as either a blind, low-vision, or color-blind person might experience it [14]. Recent extensions add a visual overlay that illustrates reading order [13]. WAVE has recently introduced features designed to help developers appreciate the problems that disabled users might experience when accessing the web. For instance, one option displays a web page in the linear order exposed by screen readers. Although such tools help developers understand how certain groups might experience the web, they miss the personal nature of accessibility and ignore the experience that many users have developed to overcome common issues.

Screening with Assistive Technology: *Screening* is the use of assistive technology to help identify accessibility issues. Henry describes the advantages and limitations of using screening techniques to evaluate accessibility [6]. The advantage is that it may help someone appreciate the experience of someone with different capabilities. A disadvantage, especially when using a complicated software program like a screen reader, is that an inexperienced user may incorrectly associate their own inability to use the tool with accessibility problems. Although valuable, screening also requires expensive screen readers that developers are unlikely to install.

Engaging Users: Mankoff *et al.* compared the results of multiple sighted developers using screening techniques with evaluation by remote blind users [8]. Developers using screen readers found many more problems than did the blind evaluators, although the problems found by blind users were quite accurate. Commonly, when blind users perform an evaluation they focus on the most problematic accessibility problems and sometimes are not able to fully evaluate a web site because the accessibility problems discovered prevent full access to the site [10]. Takagi *et al.* describes the problems of users not knowing what is not accessible to them as a challenge for IBM’s Social Accessibility project [15]. WebTrax seeks to help blind web users or experienced educators demonstrate the non-visual web browsing experience to sighted developers.

TrailBlazer is a programming by demonstration system that lets blind web users record, share and replay web tasks [2]. The recordings that they make are expressed in the CoScripter language [7], which uses pseudo-natural language commands that are easy to understand. User actions are recorded as commands like “Click on the ‘Search’ button” and “enter ‘University of Rochester’ into the ‘Search’ textbox”. TrailBlazer scripts are shareable, so a user can record themselves navigating through a web page and then share that with a developer in order to convey to how they browsed the content. Unlike WebTrax, however, TrailBlazer does not record information about all of the content that has been viewed by blind web users - it records actions that have been taken but not the content that has been read. The visualizations supplied by TrailBlazer may allow

developers follow along, but, because only one step is highlighted at a time, it can be difficult to get a global view of what the user has done.

3 Visualizing Web Browsing with WebTrax

To use WebTrax, users first load the WebTrax-enabled WebAnywhere and turn on one of the supported visualizations with an associated shortcut key. The blind user or educator driving the interface then browses web content and visualizations are added automatically within the existing WebAnywhere interface.

3.1 Basic Visualization

The basic visualization highlights the current content that is being read (Figure 2). This feature was initially created to allow WebAnywhere to work better for people with low-vision, but we have found that developers appreciate it because it makes following along easier.

3.2 Phosphor Tracking

The *Phosphor Tracking* visualization keeps elements highlighted even after they have been read in order to remind observers of the trail that users have taken through the content of the web page. This visualization was inspired by Phosphor interfaces [1], which are designed to highlight the interface components with which users have interacted and fade slowly away once they are done. These interfaces have been shown to help users remember what they've done and highlight errors.

Our Phosphor Tracking visualization reveals not only the content that has been read but also a sense of the time elapsed since the user visited each part of the page (Figure 3). Content is highlighted as it is read to the user. This highlighting slowly fades over time leaving a trail back to the start of the page.

3.3 Generating Heatmaps

Heatmaps have become a staple in usability as a way to visualize eye-tracking data [9]. Heatmaps display portions of the content where users have looked the most a warmer color (such as red) and portions of the content where users looked the least a cooler color (such as blue). These heatmaps are made possible by expensive and difficult to calibrate eye-tracking technology that records where sighted users are looking.

The third visualization that WebTrax supports is a heatmap over non-visual interactions. This visualization uses two sources of data to generate the heatmap and its color. First, any web page elements that a user has visited is colored. The color is determined based on how much of the content a user listened to before skipping past the element. For example, if the user listened to an entire heading then that heading would be colored red, if they quickly skipped past it

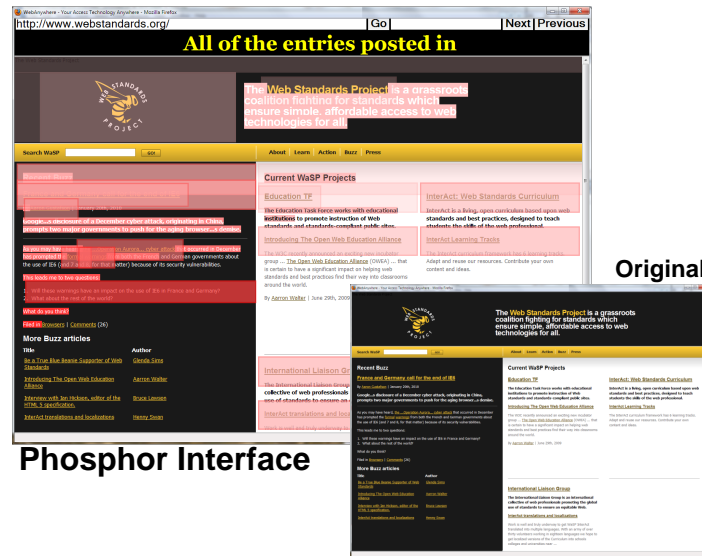


Fig. 3. The phosphor visualization of non-visual web browsing keeps the elements that users visit visible for some time and lets them slowly fade away over time. In this example, we can see that the content on the right actually comes earlier in the reading order than the content on the left because it has faded more as the user has browsed.

then it would be colored blue, and if they listened to only half of the heading then it would be colored yellow-orange (Figure 4).

The heatmap visualization is created in a transparent Flash movie that has been positioned over the portion of the page that has been read. Each point is assigned a weighted heat value, and its influence is distributed over a fixed radius. Several points can influence the displayed value for a single rendered pixel. The color for a given pixel in the bitmap is determined by the amount of influence that has been exerted over that pixel by all of the elements near it [16]. The values are distributed over a limited hue space in the HSV color representation system. The highest or “hottest” values are red, followed by orange, yellow, and blue for lower or “colder” values.

Heatmaps give observers a global picture of what content is easy to access, and which irrelevant content users must hear before finding what they want. We have found heatmaps are quite adept at identifying the templates of popular web sites - users skip quickly past template information (such as sidebars) causing them to be colored blue, while the main content areas are often colored red.

4 Future Directions

WebTrax is a promising approach for helping web developers better understand the interactions and browsing strategies of screen reader users. We plan to conduct formal users studies to evaluate the efficacy of these visualizations in helping

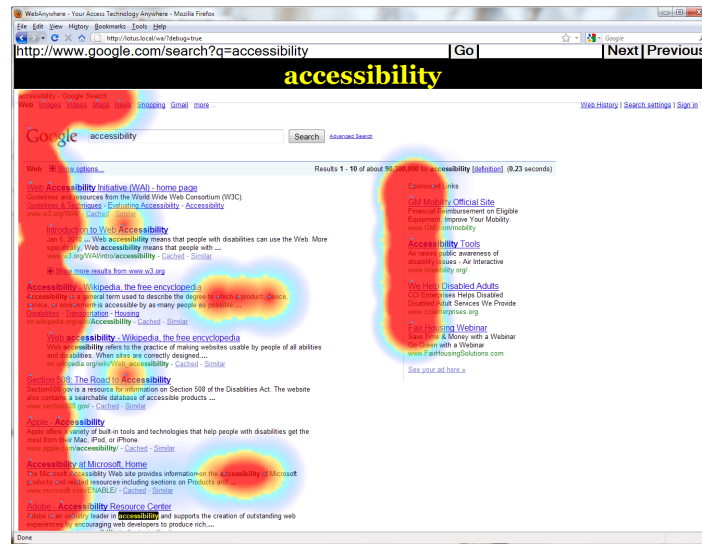


Fig. 4. The heatmap visualization of a user interacting with `google.com` search results shows all of the elements visited by screen reader users with color determined by their displayed interest that content. In this example, we can tell from the heatmap that the user was able to find the main search results, but was distracted at first by the ads on the page. The user was briefly distracted by the options near the beginning of the page, and the summaries of each search result were, for the most part, not fully read.

sighted observers better understand how blind web users are browsing through content. Particularly promising applications of this technique include helping to educate web developers and helping web developers evaluate their content.

Because WebTrax is built on top of WebAnywhere, it suffers from the same limitations of the WebAnywhere platform. Specifically, WebAnywhere does not let its users browse local files or files located behind a firewall, which means that developers who want to try WebTrax on their content will need to put the content on a publicly-facing site. These issues are not fundamental limitations of the visualizations presented here, but rather limitations of our current implementation. Many popular screen readers do not provide the programmatic access necessary to support the visualizations introduced here. We hope other screen readers will either expose these visualizations as well or open their APIs to allow third-party tools to do so.

A promising direction for future research is to aggregate the web browsing histories of multiple users together to create more comprehensive visualizations of browsing patterns and problems. The data used to generate these visualizations may also be useful for automatically fixing observed problems by, for instance, making important content that is rarely visited closer to the beginning in the screen reader's reading order.

5 Conclusion

We have presented and motivated three new visualizations that may help developers and others better understand how screen reader users are interacting with the web pages that they visit. Few tools seek to directly support a collaboration between assistive technology users and developers, but we see this as a promising new approach to teaching developers about the importance and subtleties of accessibility and usability.

References

1. Baudisch, P., Tan, D., Collomb, M., Robbins, D., Hinckley, K., Agrawala, M., Zhao, S., and Ramos, G. Phosphor: Explaining transitions in the user interface using afterglow effects. In UIST 2006.
2. Bigham, J., Lau, T., and Nichols, J. Trailblazer: Enabling blind users to blaze trails through the web. In IUI 2009.
3. Bigham, J. P., Prince, C. M., and Ladner, R. E. Webanywhere: A screen reader on-the-go. In W4A 2008.
4. Craven, T. C. Some features of alt text associated with images in web pages. *Information Research*, 11.
5. Firefox accessibility extension (2006). Illinois Center for Information Technology.
6. Henry, S. *Just Ask: Integrating Accessibility Throughout Design*. Lulu.com, London, United Kingdom, 2007.
7. Leshed, G., Haber, E. M., Matthews, T., and Lau, T. Coscripiter: Automating & sharing how-to knowledge in the enterprise. In CHI 2008.
8. Mankoff, J., Fait, H., and Tran, T. Is your web page accessible?: a comparative study of methods for assessing web page accessibility for the blind. In CHI 2005.
9. Nielsen, J. and Pernice, K. *Eyetracking Web Usability*. New Riders Press, 2009.
10. Petrie, H., Hamilton, F., King, N., and Pavan, P. Remote usability evaluations with disabled people. In CHI 2006.
11. Petrie, H. and Kheir, O. The relationship between accessibility and usability of websites. In CHI 2007.
12. Roadmap for accessible rich internet applications (wai-aria roadmap). World Wide Web Consortium (2007). <http://www.w3.org/TR/WCAG20/>.
13. Sato, D., Kobayashi, M., Takagi, H., and Asakawa, C. What's next? a visual editor for correcting reading order. In INTERACT 2009.
14. Takagi, H., Asakawa, C., Fukuda, K., and Maeda, J. Accessibility designer: visualizing usability for the blind. In ASSETS 2004.
15. Takagi, H., Kawanaka, S., Kobayashi, M., Sato, D., and Asakawa, C. Collaborative web accessibility improvement: Challenges and possibilities. In ASSETS 2009.
16. Wand, M. P., and Jones, M. C. Comparison of Smoothing Parameterizations in Bivariate Kernel Density Estimation. *Journal of the American Statistical Association*, Vol. 88, No. 422. 1993.
17. Watchfire bobby. <http://www.watchfire.com/products/webxm/bobby.aspx>.
18. Wave web accessibility evaluation tool (2009). <http://wave.webaim.org/>.