

Accessmonkey: Enabling and Sharing End User Accessibility Improvements

Jeffrey P. Bigham

Department of Computer Science and Engineering,

University of Washington, Seattle, WA 98195 USA

jbigham@cs.washington.edu

Abstract

This paper proposes the Accessmonkey Framework for collaborative web accessibility improvement. The goal of the system is to provide a common platform on which end users and developers can create, share and use scripts that specify accessibility improvements. We propose browser plugins that will automatically retrieve user scripts from a common repository and apply helpful transformations to pages visited by users. End user interfaces will allow anyone to create and share new improvements and developer interfaces will allow content creators to edit and save changes scripts make to web content. Accessmonkey seeks to enable those most impacted by web accessibility challenges to directly and independently improve them. This work is being pursued with close consultation and participation by blind web users.

Introduction

Efficient web access for blind users can be challenging. When accessed using screen reader, information encoded visually is inaccessible, functionality requiring the use of the mouse is unavailable, and complex or lengthy content lacking semantic markup is inefficient to browse. We've shown that blind users browse less efficiently as a result and avoid content that is inaccessible [2]. Extensive research has explored the benefits of transforming content to make it more accessible. These tools generally operate either fully automatically or as part of improved developer tools. Comparatively little work has enabled end users to independently improve content. Existing user and developer tools could also benefit by enabling users to share new improvement strategies, and by offering a convenient mechanism by which third-party developers could extend the diversity of existing user and developer tools. The Accessmonkey Framework seeks to address these shortcomings.

Research goals

The goals of the Accessmonkey Framework are threefold:

1. **End User Improvement** - Enable independent improvement of web content by those without programming experience using an accessible interface designed for end users.
2. **Collaboration** - Facilitate sharing of improvements among both users and web developers. Provide a forum for users to help one another.
3. **Common Platform** - Provide a common platform for accessibility improvement that will help improvements reach end users. Technology for improving web accessibility often languishes because no convenient, widely-used platform exists for releasing it.

Proposed solution

The Accessmonkey Framework will provide 1) a common platform for accessibility improvement, 2) end user interfaces that enable users to find, make and share improvements to pages, 3) developer interfaces that enable developers to find, edit and incorporate accessibility improvements into their pages, 4) browser plugins that perform the specified accessibility improvements, and 5) a repository where users can post and retrieve accessibility improvement scripts. The illustration below shows a diagram of the proposed system.

The Accessmonkey Framework will build on existing end user programming tools, such as Greasemonkey [10], Koala [7] and Keyword Commands [8], with added functionality targeted at improving accessibility. Our system will address two shortcomings of these tools: First, writing scripts to improve accessibility with existing tools either requires programming knowledge or requires the use of interfaces that are not accessible. Second, sharing improvements is inconvenient because each tool requires users to visit a separate site and manually install scripts.

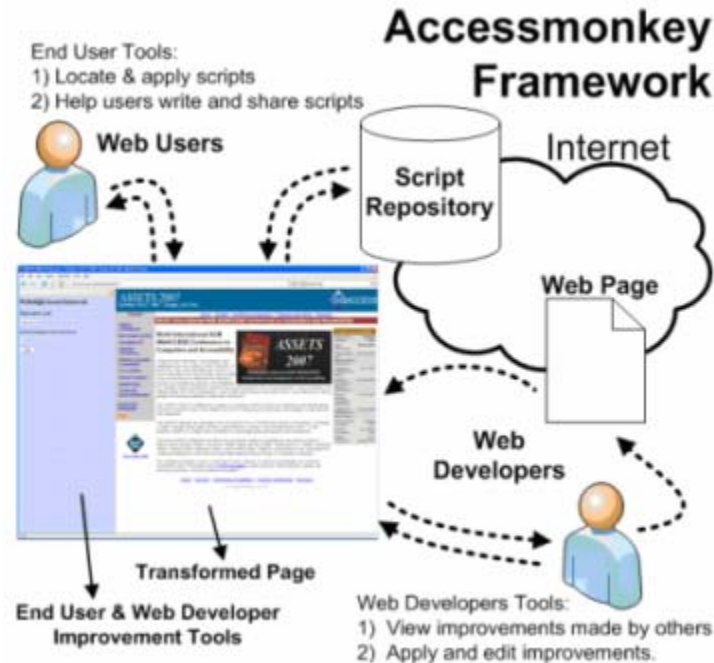


Figure 1: The Accessmonkey Framework enables web users to locate, apply, create and share scripts for accessibility improvement and enables web developers to incorporate those changes in the original pages.

End user improvement

We plan to develop interfaces that will enable blind web users who are not programmers to improve the accessibility of the web pages that they visit. Platypus [12] enables users to create Greasemonkey scripts by visually selecting items and then change their style, moving them elsewhere on the page or deleting them entirely. Keyword Commands [8] allows commands to be entered in pseudo-natural language and Koala [7] further relaxes this syntax and operates in the web domain. These extensions have limited non-visual support and lack features specific to improving accessibility.

Our interface will allow end users to improve content without programming. We are currently collecting many examples of the accessibility problems faced by blind users in order to isolate the specific transformations that should be supported by our tool. Users could, for instance, select text (using either the keyboard or mouse or through a keyword-based interface) and then specify that it should be a heading of a particular level. As other examples, users could add skip links or add list tags to information displayed as such. Future functionality could allow users to assign semantic roles to dynamic content from the ARIA [11] ontology or provide roles to arbitrary content from an ontology designed to improve accessibility [1, 13]. To develop our end user interface, we will use a user-centered design methodology.

Finding scripts

Our system will be designed to efficiently locate, install and apply user-created scripts. Current user scripting tools [10, 5, 7] do not automatically locate and apply scripts because the security of such scripts cannot be guaranteed. Scripts that send private information to a remote site or make other malicious changes are problems for social networking sites [6], and previous versions of Greasemonkey were shown to be susceptible to similar exploits [10]. Instead, existing systems require users to manually find and install scripts, offering a modest level of protection. User scripts need to be powerful yet appropriately constrained to enable arbitrary users to contribute improvements.

We will limit and rigorously secure the operations supported by our end user interfaces, and create a community rating system to help users find the best scripts. More advanced scripts that require more of the Javascript functionality will be thoroughly vetted by the community before being applied automatically.

Preliminary results

We have introduced the common platform component of Accessmonkey [4]. It extends Greasemonkey in two important ways: 1) Changes that are made to pages can be saved by developers in order to improve the original web page. 2) The system is available on multiple browsers and platforms.

We also demonstrated how Accessmonkey scripts can improve accessibility. Site- or page-specific scripts identify a common template or page element and alter it to be more accessible. For example, content is rearranged so that important information is read first and a dynamic menu is made accessible using only the keyboard. General scripts apply to all web pages. Our WebInSight script adds alternative text to web images [3], our context-focused browsing script emulates the Hearsay browser's CSurf [9] and our headings script automatically adds HTML heading tags.

Future and ongoing work

We are currently conducting focus groups with blind web users in order to study how to best create interfaces for selecting and annotating content in a way that will attract interest from the blind community, which will determine the success of this approach. We will soon begin designing, building and testing these interfaces. To identify the most useful improvements that Accessmonkey should enable blind users to provide, we are collecting examples of the accessibility challenges confronted during the everyday browsing of blind web users and isolating common problems. Finally, we are preparing to release versions of Accessmonkey for the Mozilla Firefox and Internet Explorer web browsers.

Further information can be found on our web page:
<http://webinsight.cs.washington.edu/accessmonkey/>

References

1. S. Bechhofer, S. Harper, and D. Lunn. Sadie: Semantic annotation for accessibility. In Proceedings of the International Semantic Web Conference (ISWC '06), 2006.
2. J. P. Bigham, A. C. Cavender, J. T. Brudvik, J. O. Wobbrock, and R. E. Ladner. WebinSitu: A Comparative Analysis of Blind and Sighted Browsing Behavior. In Proceedings of the 9th International ACM SIGACCESS Conference on Computers and Accessibility (ASSETS '07), 2007.
3. J. P. Bigham, R. S. Kaminsky, R. E. Ladner, O. M. Danielsson, and G. L. Hempton. WebInSight: Making Web Images Accessible. In Proceedings of the 8th International ACM SIGACCESS Conference on Computers and Accessibility (ASSETS '06), 2006.
4. J. P. Bigham and R. E. Ladner. Accessmonkey: A collaborative scripting framework for web users and developers. In Proceedings of the International Cross-Disciplinary Conference on Web Accessibility (W4A '07), 2007.
5. M. Bolin, M. Webber, P. Rha, T. Wilson, and R. C. Miller. Automation and customization of rendered web pages. In Proceedings of the ACM Symposium on User Interface Software and Technology (UIST '05), 2005.
6. T. Jim, N. Swamy, and M. Hicks. Defeating script injection attacks with browser-enforced embedded policies. In Proceedings of the 16th International Conference on World Wide Web (WWW '07), pp. 601–610, 2007.
7. G. Little, T. A. Lau, A. Cypher, J. Lin, E. M. Haber, and E. Kandogan. Koala: capture, share, automate, personalize business processes on the web. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '07), pages 943–946, 2007.
8. G. Little and R. C. Miller. Translating keyword commands into executable code. In Proceedings of the ACM Symposium on User Interface Software and Technology (UIST '06), pages 135–144, New York, NY, USA, 2006.
9. J. Mahmud, Y. Borodin, and I. Ramakrishnan. Csurf: A context-driven non-visual web-browser. In Proceedings of the International Conference on the World Wide Web (WWW '07), pp. 31–40, 2007.
10. M. Pilgrim. Greasemonkey Hacks: Tips & Tools for Remixing the Web with Firefox. O'Reilly Media, 2005.
11. Roadmap for Accessible Rich Internet Applications (WAI-ARIA Roadmap). World Wide Web Consortium, <http://www.w3.org/TR/WCAG20/>. 2007.
12. S. R. Turner. Platypus Firefox Extension. <http://platypus.mozdev.org/>. 2007.
13. Y. Yesilada, S. Harper, C. Goble, and R. Stevens. Screen readers cannot see (ontology based semantic annotation for visually impaired web travellers). In Proceedings of the 4th International Conference on Web Engineering (ICWE '04), pp. 445–458, 2004.

About the author



Jeffrey P. Bigham is a Ph.D. candidate in the University of Washington Computer Science & Engineering Department in Seattle, WA, USA. His interests center around improving the human interfaces used to access web content with a focus on improving the interfaces used by blind web users because of how inefficient access is for this group. Along with his advisor Richard E. Ladner, he started the WebInSight group at the University of Washington, which pursues research targeted at understanding challenges for web access by screen reader users and develops innovative solutions to address them. Prior to his work in web accessibility, Jeffrey pursued research in natural language processing and extracting factual information from large, unstructured resources like the web. <http://www.cs.washington.edu/homes/jbigham/>