XRAY: Inspector Tools For Designers

Leave Authors Anonymous
for Submission
City, Country
e-mail address

Leave Authors Anonymous
for Submission
City, Country
e-mail address

Leave Authors Anonymous
for Submission
City, Country
e-mail address

ABSTRACT

Examples, both in the form of web templates and snippets of code, are widely used in the creation of websites. Numerous tools have been created to facilitate developer's borrowing of code from live websites and galleries. These include in-browser developer tools, firebug, and other tools. However, little exploration has been done on the experience of designers working on partially-developed or live sites. This paper introduces XRAY, inspector tools for designers, which allow the designer to adjust fonts, colors, margins, and padding without ever needing to look at HTML or CSS. These features were designed to increase experimentation by automating consistency (percolating changes), making this technical, code-based task more approachable for designers by creating a UI similar to (??) instead of traditional inspector tools, and improving designer-developer communication by allowing designers to export a document with all changes in a format that developers will be comfortable with. Moreover, XRAY promotes the use of design systems by only suggesting styling options that exist in the current design system and highlighting where current aesthetics violate the design system. Our user study showed that experienced designers were xx% more efficient and yy% more successful when they used xray than the standard industry tools.

ACM Classification Keywords

H.5.m. Information Interfaces and Presentation (e.g. HCI): Miscellaneous; See http://acm.org/about/class/1998/ for the full list of ACM classifiers. This section is required.

Author Keywords

Web design; design systems; inspector tools; experimentation; human factors; developers; designers.

INTRODUCTION

Both programmers and web designers look at websites for design inspiration and at snippets of code for guidance. Researchers and industry professionals have made developers tools to enable these programmers and designers to learn more from these examples. However, many of these tools are geared towards those who are comfortable with code.

Currently, the status quo is (explain how it works for most designers). (Most designers use tools and processes like these to tweak existing websites (markup in sketch, emails, meeting in person, etc).

We contribute XRAY, design tools for developers, a set of novel design tools that allow the designer to adjust fonts, colors, margins, and padding without ever needing to look at HTML or CSS.

These features were designed to increase experimentation by automating consistency (percolating changes), making this technical, code-based task more approachable for designers by creating a UI similar to (??) instead of traditional inspector tools, and improving designer-developer communication by allowing designers to export a document with all changes in a format that developers will be comfortable with. First, increase experimentation by:

- Automating consistency by allowing percolating changes (A. Facilitates designer to copy styles from other elements)
- Makes the technical aspects of coding a website more approachable for designers (A. Allows designer to work in same medium as developers (as opposed to graphical tools like sketch) B. Allows designer to better understand how websites are made/visualize element style and positioning i. Margin v. padding, etc)
- Improve designer-developer communication (A. Keep track of all changes B. Can produce a document to show a programmer all changes C. Can have two people edit the same site at one)

Moreover, XRAY promotes the use of design systems by only suggesting styling options that exist in the current design system and highlighting where current aesthetics violate the design system. Second, promotes the use of design systems by:

- only showing styling options that exist in the design system (this can be overridden)
- highlighting where the current page aesthetics violate the design system, so these can be fixed or the design system can be updated.

Third, we contribute a user study showing the benefits of these features. The results can be summarized as:

- (things that we learn)

RELATED WORKS

- Use of examples in design galleries

CHI'16, May 07–12, 2016, San Jose, CA, USA

ISBN 123-4567-24-567/08/06...$15.00

DOI: http://dx.doi.org/10.475/123_4

- Patterns found after design mining
- Actual design patterns for websites
- Using snippets of code from/understanding live sites
- Other tools for developers
- Transferring CSS from one site to other
- But there are few tools for designers (list)
- Sketching tools (Hashimoto 2005)
- Helping non-programmers select code (Dinah 2011)
- Overall, designers have it really rough, and a lot of research seems to be focusing on designers who can code and coders who can design. That isnâĂŹt what reality is like: many people in industry can either code OR design.
- Why are design systems important? WhatâĂŹs their benefit? [Maybe this belongs in the intro, but weâĂŹll want to have a reference].
- Linked Editing (https://dl.acm.org/citation.cfm?id=1034566) This might not fit here, but it should give us language when we talk about editing multiple instances at the same time.

INTERFACE AND FEATURES OF XRAY

First: overall things. Is a chrome plugin? Can use on any website.

Lets you adjust and change: Fonts (family, weight), colors, margin, padding Automatically downloads Google fonts to make them available on the website. This is often a very tedious task that requires the designer to either install the fonts locally or have access to the code. You can copy and paste styles from one element to the next DoesnâĂŹt need to be same page Get an instant preview of those changes Can toggle on and off individual styles This means several sources can be pasted into same element (only really important because Web Crystal focuses a lot on this). Lets you download your changes (to give to developer) (final visual changelog) Developer now knows exactly what to do Which will result in less work for designer as well Whitespace is broken down into padding and margin. Normal redline documents only show spacing, but not whether it is margin or padding. Changes persist across page reloads Changes to symbols persist not only across page reloads but also across the whole website, even on different pages. Can have changes permeate through a whole page Many of these things help you see the design system (trees v. forest) Visualize website structure. Margin Padding is invisible and it is often difficult to know how whitespace is assigned. This tool colors the whitespace to show the structure. Gives a live audit of the design system to show elements that diverge. I have implemented this yet, but we could make this collaborative, so two people could make live edits to a website collaboratively. ItâĂŹs not very practical I donâĂŹt think, but it is shiny. A corollary to this is instead of exporting a file to give to developers, the designer can just share the session with a developer and they can have a live editor of the changes.

USER STUDY

Participants

12 students 12 professionals

Tasks

A, B, C (or) A, B

Procedure

And fill in this section too

Results

And put some graphs and stats and stuff in here

DISCUSSION (HARD TO WRITE UNTIL WE HAVE RESULTS)

Implications of results to science Implications of results to practice Limitations Further research

CONCLUSION

Discuss benefits...

Discuss harm/risks/dangers/drawbacks...

Discuss future work/room to grow...

REFERENCES

1. Kerry Shih-Ping Chang and Brad A. Myers. 2012. WebCrystal: Understanding and Reusing Examples in Web Authoring. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '12)*. ACM, New York, NY, USA, 3205–3214. DOI: http://dx.doi.org/10.1145/2207676.2208740
2. Niranjan Damera-Venkata, José Bento, and Eamonn O'Brien-Strain. 2011. Probabilistic Document Model for Automated Document Composition. In *Proceedings of the 11th ACM Symposium on Document Engineering (DocEng '11)*. ACM, New York, NY, USA, 3–12. DOI: http://dx.doi.org/10.1145/2034691.2034695
3. James R. Eagan, Michel Beaudouin-Lafon, and Wendy E. Mackay. 2011. Cracking the Cocoa Nut: User Interface Programming at Runtime. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology (UIST '11)*. ACM, New York, NY, USA, 225–234. DOI: http://dx.doi.org/10.1145/2047196.2047226
4. David Gibson, Kunal Punera, and Andrew Tomkins. 2005. The Volume and Evolution of Web Page Templates. In *Special Interest Tracks and Posters of the 14th International Conference on World Wide Web (WWW '05)*. ACM, New York, NY, USA, 830–839. DOI: http://dx.doi.org/10.1145/1062745.1062763
5. Paul Gross, Jennifer Yang, and Caitlin Kelleher. 2011. Dinah: An Interface to Assist Non-programmers with Selecting Program Code Causing Graphical Output. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. ACM, New York, NY, USA, 3397–3400. DOI: http://dx.doi.org/10.1145/1978942.1979448

6. Scarlett R. Herring, Chia-Chen Chang, Jesse Krantzler, and Brian P. Bailey. 2009. Getting Inspired!: Understanding How and Why Examples Are Used in Creative Design Practice. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '09)*. ACM, New York, NY, USA, 87–96. DOI: http://dx.doi.org/10.1145/1518701.1518717

7. Joshua Hibschman and Haoqi Zhang. 2015. Unravel: Rapid Web Application Reverse Engineering via Interaction Recording, Source Tracing, and Library Detection. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software &#38; Technology (UIST '15)*. ACM, New York, NY, USA, 270–279. DOI: http://dx.doi.org/10.1145/2807442.2807468

8. Joshua Hibschman and Haoqi Zhang. 2016. Telescope: Fine-Tuned Discovery of Interactive Web UI Feature Implementation. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology (UIST '16)*. ACM, New York, NY, USA, 233–245. DOI: http://dx.doi.org/10.1145/2984511.2984570

9. Raphael Hoffmann, James Fogarty, and Daniel S. Weld. 2007. Assieme: Finding and Leveraging Implicit References in a Web Search Interface for Programmers. In *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology (UIST '07)*. ACM, New York, NY, USA, 13–22. DOI: http://dx.doi.org/10.1145/1294211.1294216

10. Nathan Hurst, Wilmot Li, and Kim Marriott. 2009. Review of Automatic Document Formatting. In *Proceedings of the 9th ACM Symposium on Document Engineering (DocEng '09)*. ACM, New York, NY, USA, 99–108. DOI: http://dx.doi.org/10.1145/1600193.1600217

11. Charles Jacobs, Wilmot Li, Evan Schrier, David Bargeron, and David Salesin. 2003. Adaptive Grid-based Document Layout. *ACM Trans. Graph.* 22, 3 (July 2003), 838–847. DOI:http://dx.doi.org/10.1145/882262.882353

12. Andrew J. Ko and Brad A. Myers. 2009. Finding Causes of Program Output with the Java Whyline. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '09)*. ACM, New York, NY, USA, 1569–1578. DOI: http://dx.doi.org/10.1145/1518701.1518942

13. Ranjitha Kumar, Arvind Satyanarayan, Cesar Torres, Maxine Lim, Salman Ahmad, Scott R. Klemmer, and Jerry O. Talton. 2013. Webzeitgeist: Design Mining the Web. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '13)*. ACM, New York, NY, USA, 3083–3092. DOI: http://dx.doi.org/10.1145/2470654.2466420

14. Ranjitha Kumar, Jerry O. Talton, Salman Ahmad, and Scott R. Klemmer. 2011. Bricolage: Example-based Retargeting for Web Design. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. ACM, New York, NY, USA, 2197–2206. DOI: http://dx.doi.org/10.1145/1978942.1979262

15. Brian Lee, Savil Srivastava, Ranjitha Kumar, Ronen Brafman, and Scott R. Klemmer. 2010. Designing with Interactive Example Galleries. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '10)*. ACM, New York, NY, USA, 2257–2266. DOI: http://dx.doi.org/10.1145/1753326.1753667

16. Josip Maras, Jan Carlson, and Ivica Crnkovi. 2012. Extracting Client-side Web Application Code. In *Proceedings of the 21st International Conference on World Wide Web (WWW '12)*. ACM, New York, NY, USA, 819–828. DOI: http://dx.doi.org/10.1145/2187836.2187947

17. J. Marks, B. Andalman, P. A. Beardsley, W. Freeman, S. Gibson, J. Hodgins, T. Kang, B. Mirtich, H. Pfister, W. Ruml, K. Ryall, J. Seims, and S. Shieber. 1997. Design Galleries: A General Approach to Setting Parameters for Computer Graphics and Animation. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '97)*. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 389–400. DOI: http://dx.doi.org/10.1145/258734.258887

18. Vidya Ramesh, Charlie Hsu, Maneesh Agrawala, and Björn Hartmann. 2011. ShowMeHow: Translating User Interface Instructions Between Applications. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology (UIST '11)*. ACM, New York, NY, USA, 127–134. DOI: http://dx.doi.org/10.1145/2047196.2047212

19. Daniel Ritchie, Ankita Arvind Kejriwal, and Scott R. Klemmer. 2011. D.Tour: Style-based Exploration of Design Example Galleries. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology (UIST '11)*. ACM, New York, NY, USA, 165–174. DOI: http://dx.doi.org/10.1145/2047196.2047216

20. Arvind Satyanarayan, Maxine Lim, and Scott Klemmer. 2012. A Platform for Large-scale Machine Learning on Web Design. In *CHI '12 Extended Abstracts on Human Factors in Computing Systems (CHI EA '12)*. ACM, New York, NY, USA, 1697–1702. DOI: http://dx.doi.org/10.1145/2212776.2223695

21. Nishant Sinha and Rezwana Karim. 2013. Compiling Mockups to Flexible UIs. In *Proceedings of the 2013 9th Joint Meeting on Foundations of Software Engineering (ESEC/FSE 2013)*. ACM, New York, NY, USA, 312–322. DOI:http://dx.doi.org/10.1145/2491411.2491427