Allowing Responsive Web Modules

Ben Trovato
Institute for Clarity in Documentation
1932 Wallamaloo Lane
Wallamaloo, New Zealand
trovato@corporation.com

G.K.M. Tobin
Institute for Clarity in Documentation
P.O. Box 1212
Dublin, Ohio 43017-6221
webmaster@marysville-ohio.com

Lars Thørväld
The Thørväld Group
1 Thørväld Circle
Hekla, Iceland
larst@affiliation.org

ABSTRACT

Categories and Subject Descriptors
H.4 [**Information Systems Applications**]: Miscellaneous; D.2.8 [**Software Engineering**]: Metrics—*complexity measures, performance measures*

General Terms
Theory

Keywords
ACM proceedings, LaTeX, text tagging

1. INTRODUCTION
A module is an interchangeable and independent part of a program that typically has a single and well-defined responsibility [4]. Modular programming is a technique to reduce complexity and enable reusability. In order for a module to be reusable it must not assume in which context it is being used.

Responsive Web Design (RWD) is an approach to make an application respond to the viewport size and device characteristics. This is achieved by using CSS media queries to define conditional style rules. In this paper we will focus on the size responsiveness of modules.

The problem is that there is no way to make a module responsive without it being context-aware, due to media queries only being able to target the viewport. Thus, a responsive module using media queries is layout dependent and has therefore limited reusability.

The desired behavior of a responsive module is having its inner design responding to the size of *its container* instead of the viewport. Only then is a responsive module independent of its layout context.

This can be achieved with the theoretical feature *element queries* that enables conditional CSS rules by the properties of arbitrary elements. This paper presents a novel implementation of element queries in JavaScript, and discusses the new possibilities of GUI design that our implementation enables.

1.1 The Problem Exemplified
- MQ is not the solution to RWD. (MQ was not designed for RWD as the feature was released long before RWD)
- All elements adapt their inner design by the viewport width.
- Menu Example shows how MQ are broken.
- Limitations of MQ regarding font-size (em).

Media queries were designed to enable developers to conditionally design content by the media, such as using serif fonts when printed and sans-serif when viewed on a screen. Therefore, it is only applicable for RWD of non-modular static applications. In a world where no better solution than media queries exists for RWD, changing the layout of a responsive application becomes a cumbersome task.

Imagine an application that displays the current weather of various cities as widgets, by using a weather widget module. The module should be responsive, so that more information such as a temperature graph over time is displayed when the widget is big. When the widget is small it should only display the current temperature. Users should also be able to add, remove and resize widgets.

Such application cannot be built with media queries. Since the widgets can have varying size, the module cannot change design by breakpoints relative to the viewport. To overcome this problem we must change the application so that widgets always have the same size. This implies that the size of the module and the media query breakpoints are coupled/intertwined, i.e. they are proportional to each other. The problem now is that we have removed the reusability of the weather module, since it requires the specific width that is correctly proportional to the media query breakpoints.

Imagine a company working on a big application that uses media queries for responsiveness (i.e., each responsive module assumes to have a specific percentage of the viewport size). The ability to change is desired by both developers

and stakeholders, but is limited by this responsive approach. The requirement of changing a menu from being a horizontal top menu to being a vertical side menu implies that all responsive modules break since the assumed proportionality of each module is changed. Even worse, if the menu is also supposed to hide on user input the responsiveness of the module breaks since the layout changes dynamically. The latter requirement is impossible to satisfy without element queries.

Additionally, it is popular to define breakpoints relative to the font size. Media queries can only target the font size of the document root, limiting the functionality drastically. With element queries, breakpoints may be defined relative to the font size of the targeted element.

As we can see, even with limited requirements there still are significant flaws with using media queries for responsive modules.

1.2 Requirements

- Parents should decide the layout of their children, and the children should adapt their inner design accordingly.
- Valid language syntaxes (HTML, CSS, JS).

First, a solution must provide a possibility for an element to change depending on the properties of the parent element. Elements should automatically respond to changes of the parent size so that the correct design can be activated for each size.

Second, a solution must conform to the syntax of HTML, CSS, and JS so that the compatability of tools, libraries and existing projects is retained.

2. ELEMENT QUERIES

Media queries and element queries are similar in the sense that they both enable developers to define conditional style rules that are applied by specified criteria. The main difference is the type of criteria that can be used; in media queries device, document, and media criteria are used, while element criteria are used in element queries. It can somewhat simplified be described as that media queries target the document root and up (i.e., the viewport, browser, OS, device, input mechanisms, etc.) while element queries target the document root and down (i.e., elements of the document).

It is stated on the W3C's www-style mailing list [3] by Zbarsky of Mozilla, Atkins of Google and Sprehn of Google that element queries are infeasible to implement without restricting them. By limiting element queries to specially separated container elements that can only be queried by child elements, many of the problems are resolved [2, 1]. Therefore, the RICG is currently investigating the possibility of standardizing *container* queries.

Unfortunately, even such limited container queries requires significant effort to implement [1]. Atkins argues that a correct and full implementation is unlikely to be implemented, and therefore it might be wiser to pursue sub-standards that aids third-party solutions instead.

2.1 Why is a Native Implementation troublesome?

- Performance issues.
- Cite Tab Atkins of RICG (he states that it is infeasible to standardize this).

2.2 A JavaScript Implementation

- Why is this pragmatic? Compatability, no impact (performance, language) on apps that do not need responsive modules.
- Satisfies the requirements for a solution given above.
- Present Elq's API.
- Present the performance.
- Note drawbacks (but only drawbacks for added functionality!).

3. DISCUSSION AND SUMMARY OF RELATED WORK

- Performance, APIs, Features.
- The mirror functionality of Elq makes it uniquely suitable for nested modules.

4. CONCLUSIONS

- Production ready.
- Probably no standard (or not in a long time).

5. ACKNOWLEDGMENTS

This section is optional; it is a location for you to acknowledge grants, funding, editing assistance and what have you. In the present case, for example, the authors would like to thank Gerald Murray of ACM for his help in codifying this *Author's Guide* and the **.cls** and **.tex** files that it describes.

6. REFERENCES

[1] Css containment draft. An issue that discusses why a special element viewport element is needed and why a standard for CQ might be hard to push.

[2] Ricg irc log. The log where the element query limitations were discussed.

[3] W3c public mail archive. Mail thread subject: The :min-width/:max-width pseudo-classes.

[4] D. L. Parnas. On the criteria to be used in decomposing systems into modules. *Communications of the ACM*, 15(12):1053–1058, 1972.