KTH Computer Science
and Communication

Modular responsive web design

Allowing responsive web modules to respond to custom criterias instead of only viewport size by implementing *element queries*

LUCAS WIENER
lwiener@kth.se

Master's Thesis at CSC

Supervisors at EVRY AB: Tomas Ekholm & Stefan Sennerö
Supervisor at CSC: Philipp Haller
Examiner: Mads Dam

TRITA xxx yyyy-nn

Abstract

Abstract goes here.

Referat

Modulär responsiv webbutveckling

Sammanfattning ska vara här.

Contents

Chapter 1

Introduction

1.1 Targeted audience

1.2 Problem statement

By using CSS *media queries* developers can specify different rules for different viewport sizes. This is fundamental to create responsive web applications. If developers want to build modular applications by composing the application by smaller components (elements, scripts, styles, etc.) media queries are no longer applicable. Modular components should be able to react and change style depending on the given size that the component has been given by the application, not the viewport size. The problem can be formulated as: *Elements can not specify conditional style rules depending on their own size, or the size of any other element.* See the included document in section A.1 of the appendix for a more practical problem formulation.

The main international standards organization for the World Wide Web (World Wide Web Consortium, abbreviated W3C) has unofficially stated that such feature would be infeasible to implement. Some problems with implementing element queries in CSS are:

- **Circularity:** The styling of elements depend on many factors (theoretically on all other elements in the layout tree). If elements can apply styles by criterias of other elements, it will be possible to create infinite loops of styling. The simplest example of this would be an element to set its width to 200 pixels if it is under 100 pixels wide. If the element is under 100 pixels wide, the new style will be applied to the element which would make the width of the element 200 pixels. If this element would have another rule that set its width to 50 pixels if it is wider than 150 pixels, there is an infinite loop of styling. Problems like this can probably be catched during CSS parsing, but there are so many combinations of style properties that could result in similar loops that it will add a lot of complexity to the language, both for implementers and users. See the included document in section A.2 of the appendix for more examples and deeper discussions about cyclic rules.

- **Performance:** Rendering engines typically perform selector matching and layout computations in parallel to achieve good performance. If element queries would be implemented, the rendering engines would need to first compute the layout of all elements in order to decide which selectors would conform to the element query conditions and then do a new layout computation, and so on until a stable state has been reached. Far worse, since selectors now depend on layout style, this cannot be done in parallel which impacts performance heavily.

Because of the problems, it is stated that such feature will not be implemented in the near future. So it is now up to the developers to implement this feature as a third-party solution. Efforts have been made by big players to create a robust implementation, with moderate success. Since all implementations have shortcomings, there is still no de facto solution that developers use and the problem remains unsolved.

1.3 Objective

The main objective of this thesis is to develop a third-party implementation of element queries (or equivalent to solve the problem of modular responsive elements). To do this, it is needed to research existing implementation attempts in order to understand and analyze the advantages and shortcomings of the approaches. It is also neccesary to be aware of the premises, such as browser limitations and specifications that need to be conformed. In addition, research will be done about the problems of implementing element queries natively, to get a deeper understanding of how an official API would look like. There are many challenges along the way that will need to be researched and worked around. Examples of such subproblems that would need to be investigated are:

- How should circularity be handled? Should it be detected at runtime or parsetime, and what should happen on detection?
- How can one listen to element dimension changes without any native support?
- How can a custom API be crafted that will enable element queries and still conform to the CSS specification?
- If a custom API is developed, how would one make third-party modules (that uses media queries) work without demanding a rewrite of all third-party modules?

The scientific question to be answered is if it is possible to solve the problem without extending the current web standard. The hypothesis is that the problem can be solved in a reliable and performant way by crafting a third-party implementation. A reliable implementation should also enable existing responsive components to react to a specified criteria (parent container size for example) with no modifications to the components. The goal of the thesis should be considered fulfilled if a solution was successfully implemented or described, or if the problems hindering a solution are thoroughly documented.

1.4 Significance

Web developers are today limited to writing big applications in an entagled mess. In almost all other programming environments it is possible and encouraged to write applications in smaller parts (or modules). By creating modules that can be used in any context with well defined responsibilities and dependencies, developing applications is reduced to the task of simply configuring modules (to some extent) to work together which forms a bigger application. It is today possible to write the web client logic in a modular way in JavaScript. The desire of writing modular code can be shown by the popularity of frameworks that helps dividing up the client code into modules. The ever so popular frameworks Angular, Backbone, Ember, Web Components, requirejs, Browserify, Polymer, React and many more all have in common that they embrace coding modular components. Many of these frameworks also help with dividing the HTML up into modules, creating small packages of style, markup and code. One of the biggest issues keeping the modules from being truly modular is that they cannot adapt to given sizes. This makes the modules either force the client to style them properly depending on viewport size, or not being responsive. Both options are undesirable for developing larger applications. A third option would be to make the modules context aware and style themselves according to the viewport, which defeats the purpose of modules (making them not reusable).

The last couple of years a lot of articles have been written about the problem and how badly we need element queries. As already stated, third-party implementation efforts have been made by small and big players, with moderate success. W3C keep getting requests and questions about it, but the answer seems to lean towards no. An organization called Responsive Issues Community Group (abbreviated RICG) have started an initial planning regarding element queries. However, things are moving slow and a draft about element queries use cases are still being made.

Solving this problem would be a big advancement to web development, enabling developers to create truly modular components. By studying the problem, identifying approaches and providing a third-party solution the community can take a step closer to solve the problem. If the hypothesis holds, developers will be able to use element queries in the near future, while waiting for W3C to make their verdict. The outcome of this thesis can also be helpful for W3C and others to get an overview of the problem and possibly get ideas how subproblems can be handled.

1.5 Methodology

TODO: What should go here?

1.6 Delimitations

The focus of the thesis lays on developing a third-party framework that realizes element queries. All theoretical studies and work will be performed to support the development of the framework.

1.6.1 What will be done

- A third-party implementation of element queries will be developed.
- The problems of implementing element queries natively will be addressed.
- Theory about rendering engines, CSS, HTML and responsive web design will be given to fully understand the problem.

1.6.2 What will not be done

- No efforts will be made to solve the problems accompanied with a native solution.
- No API or similar will be designed for a native solution.
- UI and UX design will not be addressed, other than necessary for understanding the problem.
- No complete history of browsers, the Internet or responsive web design will be given other than neccesary.

1.7 Outline

Part I

Background

Chapter 2

Browsers

↪ *Browsers and the Internet is something that many people today take for granted. It is not longer the case that only computer scientists are browsing the web. Today the web is becoming increasingly important in both our personal and professional lives. This chapter will give a brief history of browsers and the rise of the web. It will also cover the role of browsers today, and what can be expected in the future. This section is a summary of* [13][1][5][15][12].

Before addressing the birth of the web, lets define the meaning of the concepets of the *Internet*, *Web* and *World Wide Web*. The word internet can be translated to *something between networks*. When referring to the Internet (capitalized) it is usually the global decentralized internet used for communication between millions of networks using TCP/IP. Since the Internet is decentralized, there is no owner. Or in other words, the owners are all the network end-points which means all users of the Internet. One can argue that the owners of the Internet are the ISP's, providing the services and infrastructure making the Internet possible. On the other hand, the backbones of the Internet are usually co-founded by countries and companies. Or is it the ICANN[1] organization which has the responsibility for managing the IP addresses in the Internet namespace? Clearly, the Internet wouldn't be what it is today without all the actors. The Internet lays the ground for many systems and applications, including the World Wide Web, file sharing and telephony. In 2014 the number of Internet users was measured to just below 3 billions, and estimations shows that we have surpassed 3 billions users today (no report for 2015 has been made yet). Users are here defined as humans having unrestricted acccess to the Internet. If one instead measures the number of connected entities (electronic devices that communicates through the Internet) the numbers are much higher. An estimation for 2015 of 25 billions connected entities has been made, and the estimation for 2020 is 50 billions.

As already stated, the Word Wide Web (abbreviated WWW or W3) is a system that operates through the Internet. The World Wide Web is usually shortened to

[1]The Internet Corporation for Assigned Names and Numbers

simply *the web*. The web is a system for accessing interlinked hypertext documents and other media such as images and videos. Since not only hypertext is interlinked on the web, the term *hypermedia* can be used as an extension to hypertext which also includes other nonlinear medium of information (images, videos, etc.). Although the term hypermedia has been around for a long time, the term hypertext is still being used as synonym for hypermedia. Further, the web can also be referred to as the universe of information accessible through the web system. Therefore, the web is both the system enabling sharing of hypermedia and also all of the accessible hypermedia itself. Hypertext documents are today more known by the name *web pages* or simply *pages*. Multiple related pages compose a *web site* or simply a *site* and are usually hosted from the same domain. To transfer the resources between computers the protocol HTTP is used. Typically the way of retrieving resources on the web is by using a *web browser* or simply a *browser*. Browsers handles the fetching, parsing and rendering of the hypertext (more about this in section 2.2).

2.1 The origin of the web

↪ *Since the web is a system operating on top of the Internet, it is needed to first investigate the origin of the Internet. This can be viewed from many angles and different aspects need to be taken into consideration. With that in mind, the origin of the Internet is not something easily pinned down and what will be presented here will be more of a technically interesting history. This section is a summary of* [6][8][7][4].

In the early 1960's *packet switching* was being researched, which is a prerequisite of internetworking. With packet switching in place, the very important ancestor of the Internet ARPANET (Advanced Research Projects Agency Network) was developed, which was the first network to implement the TCP/IP protocol suite. The TCP/IP protocol suite together with packet switching are fundamental technologies of the Internet. ARPANET was funded by the United States Department of Defense (DoD) in order to interconnect their research sites in the United States. The first nodes of ARPANET was installed at four major universities in the western United States in 1969 and two years later the network spanned the whole country. The first public demonstration of ARPANET was held at the International Computer Communication Conference (ICCC) in 1972. It was also at this time the email system was introduced, which became the largest network application for over a decade. In 1973 the network had international connections to Norway and London via a satelite link. At this time information was exchanged with the File Transfer Protocol (FTP), which is a protocol to transfer files between hosts. This can be viewed as the first generation of the Internet. With around 40 nodes, operating with raw file transfers between the hosts it was mostly used by the academic community of the United States.

The number of nodes and hosts of ARPANET increased slowly, mainly due to the fact that it was a centralized network owned and operated by the US military. In 1974 the TCP/IP stack was proposed in order to have a more robust and scalable system for end-to-end network communication. The TCP/IP stack is a key technology for the decentralization of the ARPANET, to allow the massive expandation of the network that later happened. In 1983 ARPANET switched to the TCP/IP protocols, and the network was split in two. One network was still called ARPANET and was to be used for research and development sites. The other network was called MILNET and was used for military purposes. The decentralization event was a key point and perhaps the birth of the Internet. The Computer Science Network (CSNET) was funded by the National Science Foundation (NSF) in 1981 to allow networking benefits to academic insitutions that could not directly connect to ARPANET. After the event of decentralizing ARPANET, the two networks were connected among many other networks. In 1985 NSF started the National Science Foundation Network (nsfnet) program to promote advanced research and education networking in the US. To link the supercomputing centers funded by NSF the NSFNET serverd as a high speed and long distance backbone network. As more networks and sites were linked by the NSFNET network, it became the first backbone of the Internet. In 1992, around 6000 networks were connected to the NSFNET backbone with many international networks. To this point, the Internet was still a network for scientists, academic institutions and technology enthusiasts. Mainly, because NSF had stated that NSFNET was a network for non-commercial traffic only. In 1993 NSF decided to go back to funding research in supercomputing and high-speed communcations instead of funding and running the Internet backbone. That, along with an increasing preassure of commercializing the Internet let to another key event in the history of the Internet - the privatization of the NSFNET backbone.

In 1994, the NSFNET was systematically privatized while making sure that no actor owned too much of the backbone in order to create constructive market competition. With the Internet decentralized and privatized regular people started using it as well as companies. Backbones were built across the globe, more international actors and organizations appeared and eventually the Internet as we know it today came to exist.

2.1.1 The World Wide Web

↪ *Now that the history of the Internet has been described, it is time to talk about the birth of the World Wide Web. Here the initial ideas of the web will be described, the alternatives and how it became a global standard. This subsection is a summary of* [2][16][3][14][4].

Recall that the way of exchanging information was to upload and download files between clients and hosts with FTP. If a document downloaded was referring to another document, the user had to manually find the server that hosted the other document and download it manually. This was a poor way of digesting information and documents that linked to other resources. In 1989 a proposal for a communication system that allowed interlinked documents was submitted to the management at CERN. The idea was to allow links embedded in text documents, to enable users to view the linked document by clicking it. A quote from the draft:

> Imagine, then, the references in this document all being associated with the network address of the thing to which they referred, so that while reading this document you could skip to them with a click of the mouse.

This catches the whole essence of the web in a sentence — to interlink resources in an user friendly way. The proposal describes that such text embedded links would be hypertext. It continues to explain that interlinked resources does not need to be limited to text documents, multimedia such as images and videos can also be interlinked which would similarly be hypermedia. The concept of browsers is described, with a client-server model the browser would fetch the hypertext documents, parse them and handle the fetching of all media linked in the hypertext.

In 1990, the Hypertext Transfer Protocol (HTTP), the Hypertext Markup Language (HTML), a browser and a web server had been created and the web was born. One year later the web was introduced to the public and in 1993 over five hundred international web servers existed. It was stated in 1994 that the web was to be free without any patents or royalties. At this time the Wolrd Wide Web Consortium (W3C) was founded by Berners-Lee with support from the Defense Advanced Research Projects Agency and the European Commission. The organization comprised of companies and individuals that wanted to standardize and improve the web.

As a side note, the Gopher protocol was developed in parallel to the World Wide Web by the University of Minnesota. It was released in 1991 and quickly gained traction as the web still was in very early stages. The goal of the system, just like the web, was to overcome the shortcomings of browsing documents with FTP. Gopher enabled servers to list the documents present, and also to link to documents on other servers. This created a strong hierarchy between the documents. The listed documents of a server could then be presented as hypertext menus to the client (much like a web browser). As the protocol was simpled than HTTP it was often preferred since it used less network resources. The structure provided by Gopher provided a platform for large electronic library connections. A big difference between the web and the Gopher platform is that the Gopher platform provided hypertext menus presented as a file system while the web hypertext links inside hypertext documents, which provided greater flexibility. When the University of Minnesota announced that it would charge lincensing fees for the implementation, users were somewhat scared away. As the web matured, being a more flexible system with more features as well as being totally free it quickly became dominant.

2.2 The history of browsers

↪ *In the mid 1990's the usage of the Internet transitioned from downloading files with* FTP *to instead access resources with the* HTTP *protocol. To fulfill the vision* *that users would be able to skip to the linked documents "with a click of the mouse" the users needed a client to handle the fetching and displaying of the hypertext documents, hence the need for browsers were apparent. Here the evolution of the browser clients will be given, while emphasizing the timeline of the popular browsers we use today. This section is a summary of* [16][17][10][9][11].

The first web browser ever made was created in 1990 and was called WorldWideWeb (which was renamed to Nexus to avoid confusion). It was at the time the only way to view the web, and the browser only worked on NeXT computers. Built with the NeXT framework, it was quite sophisticated. It had a GUI and a WYSIWYG[2] hypertext document editor. Unfortunately it couldn't be ported to other platforms, so a new browser called *Line Mode Browser* (LMB) were quickly developed. To ensure compatability with the earliest computer terminals the browser displayed text, and was operated with text input. Since the browser was operated in the terminal, users could log in to a remote server and use the browser via telnet. The code that the two browsers shared was in 1993 bundled as a library called *libwww*. The library was licensed as *public domain* to encourage the development of web browsers. Many browsers were develop at this time. The *Arena* browser served as a testbed browser and authoring tool for Unix. The *ViolaWWW* browser was the first to support embedded scriptable objects, stylesheets and tables. *Lynx* is a text based browser that supports many protocols (including Gopher and HTTP), and is the oldest browser still being used and developed. The list of browsers of this time can be made long.

In 1993, the *Mosaic* browser was released by the National Center for Supercomputing Applications (NCSA) which came to be the ancestor of many of the popular browsers in use today. As Lynx, Mosaic also supported many different protocols. Mosaic quickly became popular, mainly due to the intuitive GUI, reliability, simple installation and Windows compatability. The leader of the team that developed Mosaic, Marc Andreessen, left NCSA to start Mosaic Communications Corporation.

[2]What You See Is What You Get is a classification that ensures that text and graphics during editing appears close to the result.

Chapter 3

Web development

3.1 From documents to applications

3.2 Responsive web design

3.3 Modularity

Chapter 4

Modular development

4.1 Web Components

Part II

Theory

Chapter 5

Rendering engines

Chapter 6

Element queries

Part III

Third-party framework

Chapter 7

Analysis of approaches

7.1 Current implementations

Chapter 8

API design

Chapter 9

Implementation

Part IV

Result

Chapter 10

Discussion

Bibliography

[1] Dave Evans. “The Internet of Things”. In: (Apr. 2011). URL: http://www.cisco.com/web/about/ac79/docs/innov/IoT_IBSG_0411FINAL.pdf (visited on 02/12/2015).

[2] *Gopher (protocol)*. Feb. 2015. URL: http://en.wikipedia.org/wiki/Gopher_(protocol) (visited on 02/24/2015).

[3] “History of the Internet”. In: (). URL: http://www.webdevelopersnotes.com/basics/history_of_the_internet.php3 (visited on 02/24/2015).

[4] “History of the Internet”. In: (Mar. 2009). URL: http://www.historyofthings.com/history-of-the-internet (visited on 02/24/2015).

[5] *Hypermedia*. Aug. 2014. URL: http://en.wikipedia.org/wiki/Hypermedia (visited on 02/12/2015).

[6] Gary C. Kessler. “An Overview of TCP/IP Protocols and the Internet”. In: (Nov. 2010). URL: http://www.sci.brooklyn.cuny.edu/~parsons/courses/3120-fall-2012/notes/tcp-ip-notes.pdf (visited on 02/17/2015).

[7] Timothy B. Lee. *40 maps that explain the internet*. June 2014. URL: http://www.vox.com/a/internet-maps (visited on 02/24/2015).

[8] Barry Leiner et al. “Brief History of the Internet”. In: (Oct. 2012). URL: http://www.internetsociety.org/sites/default/files/Brief_History_of_the_Internet.pdf (visited on 02/11/2015).

[9] *libwww*. Jan. 2015. URL: http://en.wikipedia.org/wiki/Libwww (visited on 02/25/2015).

[10] *Line Mode Browser*. Dec. 2014. URL: http://en.wikipedia.org/wiki/Line_Mode_Browser (visited on 02/25/2015).

[11] *Mosaic (web browser)*. Feb. 2015. URL: http://en.wikipedia.org/wiki/Mosaic_(web_browser) (visited on 02/25/2015).

[12] *OED Online*. Dec. 2014. URL: http://www.oed.com/ (visited on 02/12/2015).

[13] internet live stats. *Internet Users*. Feb. 2015. URL: http://www.internetlivestats.com/internet-users/ (visited on 02/12/2015).

[14] "The World Wide Web (WWW) basics and fundamentals". In: (). URL: `http://www.webdevelopersnotes.com/basics/the_world_wide_web.php3` (visited on 02/24/2015).

[15] W3C. "About The World Wide Web". In: (Jan. 2001). URL: `http://www.w3.org/WWW/` (visited on 02/12/2015).

[16] *World Wide Web*. Feb. 2015. URL: `http://en.wikipedia.org/wiki/World_Wide_Web` (visited on 02/24/2015).

[17] *WorldWideWeb*. Feb. 2015. URL: `http://en.wikipedia.org/wiki/WorldWideWeb` (visited on 02/25/2015).

Appendix A

Resources

A.1 Practical problem formulation document

TODO: Should this be in the real document instead of appendix?

A.2 Practical cyclic rules discussion document

TODO: Should this be in the real document instead of appendix?