Interpreting HTML

B206

September 22, 2011

Det Teknisk-Naturvidenskabelige Fakultet

Aalborg Universitet

Basisuddannelsen

TITLE:

Interpreting HTML

PROJECT PERIOD:

P0, september 1st - 23rd, 2011

PROJECT GROUP:

B206

GROUP MEMBERS:

Andreas Kjær Andersen

Dennis Nicolaj Biil

Jens Møller Kursch

Johan Leth Gregersen

Mikkel Sandø Larsen

Morten Møller Jakobsen

ADVISOR:

Saulius Samulevicius

COPIES: 8

REPORT PAGECOUNT: 20

TOTAL PAGECOUNT: 29

SYNOPSIS:

This report focuses on browser standards and the interpretation of HyperText Markup Language (HTML) and Cascading Style Sheets (CSS). Our focus has been on the current major browsers: Mozilla Firefox, Google Chrome and Microsoft Internet Explorer. We seeked to determine how well HTML-standards were met in websites. To do this we examined statistics for the amount of standards compliant websites as well as editors and CMS' ability to create such. We also communicated with two proffesionals to get their input. To know the span of the problem we have checked browsers ability to update in the past and looked at the coming versions of HTML and CSS. Our findings were that only a very low amount of developers abide by the standards, even though browsers implement them ”equally” well. Due to the short timespan for the report we were unable to determine a reason for this.

Contents

Chapter 1

Introduction

In this chapter we give an introduction to the basic subjects of our report.

1.1 HTML

The HyperText Markup Language is referred to as HTML and is one of the core technologies for building web pages.

It started out with a physicist named Tim Berners-Lee who was a contractor at CERN. He proposed a system to make it easier to keep track of big projects. He got his way and the first description of HTML was publicly available in 1991. However at this point it was simply just used by scientists and researchers to share information [1].

HTML 2.0 was published in 1995 as the Internet Engineering Task Force (IETF) Request For Comments (RFC), meaning they wanted feedback, and therefore it was now available for everyone to work with. It was quickly picked up by the majority of internet programmers and already in 1997 it was published as a World Wide Web Consortium (W3C) recommendation [3]. As the market grew larger and more people started using the internet in search for knowledge, presentation became a much bigger factor in making websites. The same year as HTML 3.2 was published HTML 4.0 was published as well. It offered better support for multimedia options, scripting languages and were far more accessible to all users. HTML 4.0 also featured a new supporting presentational langauge, Cascading Style Sheets (CSS). Browser support was undertaken earnestly by Microsoft in Internet Explorer (IE) and the IE5, which was the market-leading browser at the time, had excellent support for the new attributes that was launched in HTML 4. Netscape's Navigator 4.7 however, had severe problems even with the most basic CSS. The greater focus on presentation came at the cost of structure and compatibility and this meant that some pages were not accessible for users who didn't have the 'right' browser or setup [23].

This was soon a big problem for everyday people, especially with the internet being accessible through new types of devices; PDAs, phones etc. Therefore the W3C had a desire to make HTML strictly structural, and this was the origin of eXtensible Markup Languge (XML). As one out of many, HTML is based on Standard Generalized Markup Language (SGML)[2] which was not intended for people. SGML is known as a metalanguage, because it is a language used to define other languages and in order to make HTML strictly for creating structure, you want to get as close to this metalanguage as possible. Around the beginning of the 21st century the XML format was launched. It was just a simplified version of SGML, to make SGML available for web developers. XML allows you to create your own tags and attributes so they are suited to the type of work or document that you are working on. This was done so that it was possible to make eXtensible HyperText Markup (XHTML), and this is also why XHTML is often called a subset of XML. This means that you actually write in XML code, but you are restricted to a set of predetermined elements. This gives the web developer the benefits of XML, which ensures that the page is accessible by far more standard browsers [22].

In 2008 HTML5 was published and although the syntax of HTML5 has great resemblance to a lot of SGML an attempt has been made to "cut the cord" on SGML, and try to make a separate "HTML" serialization in addition to an alternative XML-based HTML5 serialization. HTML5 is by far not completely developed yet, and their goal is to have it finished sometime in the year 2014.

1.1.1 How does it work

HTML coding is written in the form of HTML elements which consists of tags, which normally come in pairs of 2, a start tag and an end tag. In-between the start tag and the end tag, the author can put additional tags. Some of these tags represent structure and others represent text, pictures, videos or others. Some tags don't have an end tag. An example of this is <br>, meaning a line break. In the beginning of HTML, it only supported a few simple tags, like <h1>Headings</h1> or the before mentioned <br>, but as the versions pass by, we got more different and more complex tags, that are more complex to use [19].

1.2 CSS

Cascading Style Sheets or CSS for short is a programming language used for describing the general look and properties for markup languages. It is also used for HTML. There are several good reasons to use CSS, especially if you're creating more than one HTML document with the same style. Creating a CSS for a webpage allows the programmer to only have to do the design for the page once, no matter how many sub pages there are. Reoccurring styles and properties can be stored in the CSS file, and applied to any number of HTML files by putting a reference in each HTML document to the CSS file. Besides not having to create the same codes for design and properties more than once, there is also less total code. Thus it becomes much easier to manage the content of each HTML page when you only have actual content to manage in the HTML documents. On the design side it's very easy to change the design, because changing the CSS will change every

HTML document linked to it. (Note: CSS can be embedded in the HTML file(s), and doesn't necessarily have to be a separate file[20].) For the home user CSS reduces the amount of bandwidth used because the CSS file only has to be downloaded once in order to work for an entire website. Therefore CSS is not only a benefit for developers. The World Wide Web Consortium (W3C) whom defines CSS describes it in these words[21]:

- **CSS** stands for **C**ascading **S**tyle **S**heets
- Styles define **how to display** HTML elements
- Styles were added to HTML4 **to solve a problem**
- **External Style Sheets** can save a lot of work
- External Style Sheets are stored in **CSS files**

1.2.1 How does it work?

In CSS you use selectors to define what you are applying rules to. You then have the properties and values to play with. For example you could write

```
h1 {background-color:#FFFFFF;}
```

This piece of code contains a selector (h1 which is header 1,) a property (the background color), and the value of this property (#FFFFFF which is white). This summary of this line is that every header 1 element will have it's background-color property set to white.

These 3 classes allow the developer to change pretty much anything from the CSS, but there is one more class to mention, the pseudo-class. This class allows for changes under certain conditions. Expanding on our previous example, you could write:

```
h1:hover {background-color:#FFFFFF;}
```

Applying :hover to h1 means that the background color only changes to white when you hover over the text with your mouse. This is a very practical addition to the CSS as this allows creating many different selectors under certain conditions. After creating your set of rules and definitions, all you have to do is tell all your HTML files where to find the CSS.

1.2.2 About CSS

The idea of something like CSS has been around for a long time, but it wasn't until approximately 1993 people really started working on how it would be possible to create something like it [7]. There have been earlier examples of standards resembling CSS, for example SGML [2] (Standard Generalized Markup Language) CSS was developed by W3C [4] in order to create a standard for style sheets. It wasn't the first of its kind, but it has been accepted by the community and has become the most used style sheet language. It was originally released December 1996, and was updated May 1998 to version 2 (later edited into 2.1) which is still in use today. While this sounds simple, the use has never been simple since no browsers fully support all the features of CSS2. When CSS1 was still in use, the first browser (Internet Explorer, Macintosh Edition) to support CSS completely wasn't ready until March 2000.

1.3 Browsers

1991 Timer Berners-Lee released the first browser called "WorldWideWeb". Only 2 years after, in 1993, Mosaic was released, it had a GUI that the browsers today still share traits with, and example is the address bar. Netscape was released in 1994, it was the first commercial browser [14]. It could show the text of a web page before it was fully loaded, which was new and innovating. It gained so much popularity that in 1995, 90% of the internet users used Netscape[18]. IE were released in 1995, it was easy to get since the instalation was handled automatically. In 2002 IE had 85% of the browser users[29]. Mozilla foundation released Firebird in 2002, it was the first open source browser. In 2004 Firebird changed name to Firefox and had 20% of the internet users using it[12]. Google released their browser Chrome in 2008. It supported 48 different languages from the beginning and gained many users all the way from the start. Chorme was updated automatically which totally removed updating from the hand of the user [18].

1.3.1 How does a browser work?

Browsers recieve information in HTML, CSS and often some kind of scripting language, typically javascript. The browser interprets the code and places the elements of the text accordingly. CSS brings even more control of the design of the web page while javascript preform actions and calculations.

1.3.2 Which browsers do we look at?

We decided to look at the browsers which is the most used at the market today, look at the list below and table 1.1. The numbers we use are from August 2011 by W3Schools[29]. Mozilla Firefox (40,6%), Google Chrome: (30,3%) and Microsoft Internet Explorer (22,4%).

Name	Created by	Engine	Released In	Open Source
Firefox	Mozilla	Gecko	2002	Yes
Chrome	Google	Webkit	2008	Yes
Internet Explorer	Microsoft	Trident	1995	No

Table 1.1: Browserinformation

Mozilla Firefox
We'll start with Firefox since it's the most popular browser today. Mozilla

Firefox started out as a open source project and still is today. Mozilla says that their mission, as a non-profit organisation, is to make the web better for everyone[11]. Mozilla strives to build rich online communities stating *"We believe in the power and potential of the Internet and want to see it thrive for everyone, everywhere."*[13].

Google Chrome
Google had the goal of making a browser that was up to date with todays use of the internet. Google wanted to make chrome as fast as they could. In the progress they created a project called v8, this project was designed to make javascript run as fast as possible in the Chrome browser. Google also came up with the idea to run each tab in the browser as an individual process so that if one tab crashed the rest of the tabs would still be up and running [24].

Microsoft Internet Explorer
Microsoft Internet Explorer is the third on our list of popular browsers. Microsoft was one of the early players on the browser market. IE comes with the Windows operation system since Microsoft is developing both. But with today's free browser choice, the first thing you do, when opening a newly installed windows machine is to choose which browser you want to use.

1.3.3 Browsers and standards

Browser maker Opera is doing an ongoing study [8] that aims to catalog the structure of the internet, to get some insight into it. During this research project they created the Metadata Analysis and Mining Application (MAMA), a tool that crawls the web and indexes the markup and scripting data from approximately 3.5 million pages. See table 1.2.

Study	Date	Passed Validation	Total Validation	Percentage
Parnas	Dec. 2001	14,563	2,034,788	0.71%
Saarsoo	Jun. 2006	25,890	1,002,350	2.58%
MAMA	Jan. 2008	145,009	3,509,170	4.13%

Table 1.2: Validation pass rate studies

The reason this study is interesting is that MAMA through the W3C's validation tools have tested how many pages actually conform to the standards from W3C [9]. The 4.13% is definitly an improvement. The MAMA study also shows that out of the pages that was displaying a W3C validation badge, only 50% of them could actually validate.

Table 1.3 shows that a lot of the websites today are not using W3C standards. Maybe this is due to the use of browser specific code, since the programmers want their pages to work in the company's choice of browser. Opera analyzed page metatags to see if there were any correlation between editing tools and validation rate. The most successful program is Apple's iWeb which validates in 81.91% of the occurrences. The lowest is Microsoft Frontpage which only validates in 0.55% of the occurrences.

Editor	Total URLs In MAMA	Quantity Passing Validation	Percentage Passing Validation
Apple iWeb	2,504	2,051	81.9%
Adobe Dreamweaver	5,954	205	3.4%
NetObjects Fusion	26,355	802	3.0%
Adobe GoLive	41,865	1,086	2.6%
IBM WebSphere	32,218	626	1.9%
Microsoft MSHTML	40,030	518	1.3%
Microsoft Visual Studio	22,936	272	1.2%
Claris Home Page	6,259	48	0.8%
Adobe PageMill	15,148	100	0.7%
Microsoft Frontpage	347,095	1,923	0.6%
Microsoft Word	24,892	154	0.6%

Table 1.3: Validation pass rates relating to editors

From table 1.4 we see that using a CMS (Content Management System) doesn't really help the on the validating either, Typo validates 12.7% of the time, Blogger does 0.3%. This might be because that most CMS isn't coded for standards.

CMS	Total URLs In MAMA	Quantity Passing Validation	Percentage Passing Validation
Typo	18,067	2,301	12.7%
WordPress	16,594	1,494	9.0%
Joomla	34,852	2,248	6.5%
Blogger	9,907	30	0.3%

Table 1.4: Validation pass rate relating to CMS

The validator can help produce valid pages, by catching errors and typos. But it won't help making the code prettier.

Chapter 2

Problem

In the following chapter we define the problem we will be working on through the report.

2.1 Introduction

Today the Internet is one of the most used mediums. You meet people over the internet, you find information, you are social and you shop on the internet. You do a lot of things on the internet, and it is often a part of everyday life. In order to see content on the Internet, you use a browser that requests a list of resources from the servers in order to display the page. Some of the code is HTML and CSS, that turns into something visual such as text setup and image placement. When using browsers there are many different choices, for example those we've mentioned, which are the most commonly used browsers: IE, Firefox and Chrome. They're all developed by different companies who have different approaches on how HTML and CSS is processed and displayed to the user. This means that whatever browser the user is using to display content from the internet,the browser has a different way to display content than other browsers. The result is that sometimes browsers might not be capable of showing the content correctly if at all. This can be a problem for the users if they use a browser which can't view the content they are looking for. This is why we are going to look into todays web standards and how they display the content to the user. We will also take a look at which browsers are up to date considering the latest standards and implementations. The result of the different browsers is that a lot of extra coding is needed to display functions and content identically in every browser. This ultimately means extra work for programmers who also need to keep themselves updated on which functions work and which do not, as new versions of the browsers are developed, and they have to try and keep the code compatible with old browser versions. We will also examine if following the standards will make a website future proof.

2.2 Problem definition

When one of today's browsers parses HTML that does not conform to standards it will attempt to interpret the deviations. This eventually means multiple ways to interpret the same data. With several versions of HTML in use and a range of browsers supporting them all, it is unlikely that non-standards-compliant content is displayed similar to all end users. In this report we will examine whether these different interpretations can be avoided by developers conforming to standards. We will also examine the effect updating browsers has on the issue.

Chapter 3

Analysis

In this chapter we analyse the problem by taking a look at browser updating, HTML5, CSS3 and how the latter two interact with eachother. Furthermore we introduce opinions of two professional webdesigners, and how they percieve the problem. In the end we discuss forced standards.

3.1 Browser updating

For the common user, the updates that are installed on their computer, are providing an updated (and idealy better) browser and possibly some new looks. Most users are not aware of what has been done codewise, or if the browser has been optimizing in ways other than what you visually can see. The World Wide Web Consortium (W3C) has stated some standards for every website to follow, but a browser also needs to be able to follow this standard. So if a new standard for websites is released browsers should update accordingly. As mentioned, we are focusing on three major browsers; IE, Firefox and Chrome. We have taken a closer look at the update schedule of these browsers. IE has been a part of any Microsoft Windows installation from Windows 98 to Windows Vista. But of the 3 major browsers it's the least used one. This could partly be due to its update schedule. IE has the least frequent update rate of the three browsers. The updates for IE are quite big and change many things in one single update at a time. Firefox is the biggest and most commonly used browser of the 3 major browsers. It seems likely that one of the keystones to its browsers popularity is the frequency of its updating and its ability to keep up with the W3C standards. With the updates, the browser feels faster and more stable for the everyday user. Updates are not noticed when using Firefox, since most people have enabled the browsers ability to automatically download and install updates. In this way, the end user is not interrupted by the browser having to update and thus updates are not discarded as a distraction. Chrome is the second most popular browser and it has the highest rate of updates. Just as Firefox, Chrome uses an automatic update system that provides multiple updates every day. Chrome does however not provide an option to turn this off, which Firefox does. This means that updates are even less intrusive.

3.2 HTML5

HTML5 is the newest branch in the HTML standard history[28]. Development of the new standard started in 2004 and is currently expected to finish in 2014. The main focus of HTML5 has been to create a collective set of standards for client-side web development. Since the first versions of HTML, which were released in the early 1990's, there has been an almost constant attempt to provide content to webpages that is not part of standard HTML. Some of these attempts have resulted in technologies such as Adobe Flash and Microsoft Silverlight, but has also been the spawning point of CSS, Javascript and thus Ajax. It is the goal of HTML5 to create one standard for the most generic web-content. HTML5 brings a range of changes/new features to previous versions of the standard [28]. Below we will focus mainly on some of those that help standardize web-development.

3.2.1 Transitioning from SGML to HTML

Discarding SGML (Standard Generalized Markup Language)[2] is the first sign of HTML5 being a milestone in the history of the standard. So far HTML has been based on SGML which is an ISO standard for markup languages and as such, HTML has been within the boundaries of a well-defined syntax. The disadvantage to this however has been the numerous ways of defining browser mode through doctype. Several definitions leads to several implementations which leads to a different result in different browsers. HTML5 only specifies one doctype which is <!doctype html> [17], implying that a webpage is written in standards compliant HTML5. HTML5 will include the Document Object Model (DOM), which is an Application Programming Interface (Or API for short), used for interacting with objects within HTML and XML documents. It defines the logical structure behind the documents and the way a document is accessed and manipulated. API is a set of rules and specifications that programs can follow to communicate with each other .HTML5 will also include XHTML5 as part of the HTML5 standard. Documents will be defined in terms of the DOM-tree of objects in the document. Two standards-compliant similar HTML5 and XHTML5 documents should produce the same DOM and thus the same webpage.

3.2.2 Introducing new elements

Some of the several new elements in HTML functions as a standardization of content [28]. But aside from that they also provide general methods of including non-standard (text/image) content in webpages. There are additional new elements, so the following should also be viewed as an introduction to the idea behind HTML5.

The math element: HTML5 introduces MathML as a part of the HTML definition, meaning that mathematical equations and the like can be written inline as html elements. This removes the need for including images or attempts to build equations using legacy HTML. The inclusion of MathML in HTML is done with some modifications, since MathML is based on XML, which HTML is not. MathML is not fully integrated in XHTML either, since that would require embedding XHTML in MathML tags, which is currently not allowed.

The svg and canvas elements: In HTML5 it is possible to include dynamically created graphical objects using either the svg or the canvas element, which are both new to HTML5. Svg is drawn by specifying elements with attributes and let the browser render these. With a page visible a svg would be manipulated through the DOM, since it is parsed as part of the object tree. The contents of the canvas element is not part of the DOM, instead this element is drawn using a javascript API. Canvas is intended for graphics that require a higher refresh rate such as games or displaying result visually.

The audio and video elements: These have been two of the most widely discussed elements of HTML5 and still are. The audio and video elements allow for simple inclusion of audio and video files. Developers of webpages would no longer have to choose between various solutions for audio/video (such as Adobe Flash) as it would be up to the developers of browsers to implement playback of the content. It is possible to provide several sources for one audio or video element allowing the browser to select the one it is able to play. The issue is however that no format (neither for audio nor video) is supported by all browsers.

3.3 CSS3

Firstly it should be noted that the CSS3 standard (just as HTML5) is still a work in progress. CSS3 is meant to replace CSS2 as the standard for the layout of webpages. CSS3 includes a wide range of new attributes for text, images and other web-content. This will however not be the primary focus of our CSS3 description. Instead we will focus on how these additions help to generalize web-development. Legacy CSS standards have been limited to setting simple attributes of elements such as font, color, size and border. More advanced layout options could only be achieved using images and/or various hacks. CSS3 provides a whole new range of options in order to reduce the amount of images used in webpages. These options include text-effects, transformation and animation which in legacy CSS would be handled by javascript. The responsibility to implement these commonly used features has been moved from web developers to browser developers[26]. This seems to be the idea of CSS3. Taking some of the hacks created in CSS2 and creating a standards compliant version in CSS3 for browsers to implement. Some examples of this are:

- Elements can now have a border with rounded corners, instead of supplying separate images for each corner and dealing with the hassle of making all browsers position the images similar.

- CSS3 defines a new property called box-sizing by which you can specify if an elements size is in- or exclusive it's border.

3.3.1 Selectors in CSS3

CSS3 also introduces a whole new set of selectors that build on those of CSS1 and CSS2[15]. Mainly these are used to select the last item in a collection, e.g. the last child of a parent element or the last sibling of an element. A negation selector has also been introduced - allowing selection of elements that does not apply to a selector. Selectors ease web-development since they allow developers through CSS to specify which elements should apply a specific style without having to apply the style to the actual element via HTML. The implementation of CSS3 selectors in major browsers is close to standard; however some are not implemented in all browsers. Only one of the new selectors is yet to be implemented in any major browser; the negation selector[27]. Implementation of new CSS3 properties is major browsers is far from complete. Many properties are not implemented in any browsers yet and some are only applicable using certain prefix; In Mozilla Firefox an animation property is applied with -moz-animation instead of the CSS3 standard -moz-. All major browsers use similar constructs for many properties.

3.4 Browser updating vs HTML5 and CSS3

If we look at the updating towards HTML5 and CSS3, there have been made a test. To see how many test points each browser gets when tested with HTML5 . As of 23th of August, 2011 Chrome scores 341 points, Firefox, 313 and IE only 141. The test points define how well the browsers support HTML5. Tests have already been carried out on the newest development versions of Firefox (version 7.0 beta) and Chrome (version 14.0.835) and they still score the same as they did on the last few updates. The scores are slowly increasing as the browsers are updated though. So with Chromes small updates nearly every day, they are steadily updating towards HTML5. Firefox are taking bigger but slower leaps towards the support of HTML5, as they don't update as frequently as Chrome. IE takes the biggest leaps in the support of HTML5: the previous version (IE 8) only had 41 test points while the upcoming update (IE 10) scores 300 of 450 points. So if we look again on the updating schedule, we can see that Firefox and Google Chrome are keeping up with the new HTML5 standards. IE doesn't support many of the features, but with IE we were able to find that the upcoming IE 10 things have improved at lot. Summarizing, we can see that we are slowly moving towards support of HTML5 and CSS3. Chrome and Firefox are only 100 test points away from full support of HTML5[25].

3.5 Professional opinions

We have asked two developers about their opinions about web standards. Anders Kursch works for TV2, a danish tv station, as a web developer. Jonas Høiby works for Codehouse, a danish webdeveloper company, as an IT consultant and web developer. We have asked them about their stance on working with the HTML standards. One of the questions was: "Do you use the standards when you develop web sites?"[1] Anders Kursch answered: "With large scale web pages it is almost impossible to use the strict standards (XHTML or HTML4 strict). Officially we use HTML4 Transitional, but it isn't something which is applied upon the HTML guidelines." Jonas Høiby answered: "Yes, I use every standard for both XHTML and HTML through out the web projects, and the same with CSS2 and soon CSS3. We do this because it is easier to make the design appear identical in different browsers." We asked them both: "Why do you need to have different browsers in mind, when you develop web sites?" Anders Kursch replied: "You need to have the different browsers in mind, because the browser market is so different. Most of the danish users use IE and many of them run older versions instead of IE8 or IE9. There are four independent browsers; Firefox, Chrome, Safari and Opera, some build upon different engines with alternating rendering properties and therefore the browsers show the content differently and we need to develop solutions which work in all of them." Jonas Høiby replied: "We're all different and we use the internet in different ways. That is why we use different browsers, but designers want the users to see their design identically in every browser. All websites should work in every browser, following the development of the browsers and still be functional in the older versions. It is simply to reach out to the biggest audience possible. This is only posible if it (the webpage) works in more than one browser." The answers show us that it's a mindset of whether or not the developers are following the standards. It may require an extra effort to write HTML and CSS that follow the standards and in large scale projects it may be even harder.

3.6 Introducing XHTML

XHTML is in most ways the same as HTML. It has all of the same features and the webpage in XHTML shouldn't look different than the one in HTML. But as indicated by the name Extensible Hyper Text Markup Language there is more to XHTML. The difference between HTML and XHTML is that XHTML is stricter, because it is written on the metalanguage XML instead of the most common SGML. Bad code that browsers might parse as intended with HTML won't necessarily work with XHTML, which means the coder is forced to use proper coding. XHTML is a recommended standard as of January 2000 by W3C[5].

Based on our findings in section 1.3.3, we see that only 4,13% of the pages were compliant, and out of the pages saying they were completely true to the specifications, only 50% actually were.

The fact that so few pages are compliant with the commonly accepted standards causes a lot of problems both in terms of cross-browser compatibility and coding. It would be ideal if everyone used the same standard, so every single webpage is compliant with every single browser. XHTML would be

[1] Note: Although the mails has been marked as quotes, we have translated them for this report, as the correspondance was communicated in danish

very ideal for this purpose. The benefits are many, but unfortunately there will be some cons too by forcing one standard on people. We have tried to look at both sides and cover the pros and cons of a forced standard.

3.6.1 Pros

There are many benefits of forcing XHTML to be the standard used to code in, and you can divide the pros into two major groups[6]:

1. Easier development and maintenance; this would mean that you would have to spend less time on your webpage. Browsers interprets invalid code differently, but if we had one standard, all the browsers had to do, was to support that one standard. It would then be possible for all browsers to view every webpage (See chapter 4).
2. Accessibility; this means that the page be viewed by more users, and the purpose of creating a website is often getting people to view it[16].
    - If we had only one standard, a website would be readable to more people.
    - A common problem with developers not using standards is that search engines like Google and Yahoo can't interpret the page properly, even though the browser might. If we had a standard this wouldn't be a problem.
    - Not following standard also creates a problem for newer devices like cellphones and tablets etc.
    - Make it easier for the programs aiding hearing or visually impaired people to experience the page.

3.6.2 Cons

While there are many benefits of forcing XHTML, there will be disadvantages too. Those listed are all temporary and might only be problems for a while:

- First of all, users dependent on HTML editors like Microsoft FrontPage and Adobe Dreamweaver will have problems living up to the XHTML standards. These programs can help the inexperienced coder to create a webpage that might get to look decent, but probably will have flaws and mistakes. For example Microsoft FrontPage, an outdated but still used program creates bad HTML code unless you specifically ask it to use XHTML See table 1.3.
- Already existing webpages will not be shown correctly if they do not abide by the XHTML standard. This means that both professional coders and the home user with a personal webpage will have to recode webpages to live up to the XHTML standard. Mark Rogers, CEO of PowerMapper Software says that it is not unlikely to find 100 mistakes on a page on a corporate website, and some corporate websites haves up to 100.000 pages.

- Forced XHTML can create expenses for people who can't update their webpages themselves.
- Programmers who use other standards for coding webpages will have to learn XHTML in order to keep coding webpages.

These disadvantages might be overwhelming, but they might only last during the transition from HTML to XHTML.

Chapter 4

Testing

In this chapter we have examined how it is possible to have old browsers interpret HTML5, even if they don't support it. To do this we apply javascript.

4.1 HTML5 code example

The idea behind HTML5 is to make the code more readable by coders[28], search engines and screen readers. In this example we are using the following elements[28] (3.1 New elements):

- header
- footer
- nav
- article
- hgroup

All of the elements are easy to understand except for one, you can read them and guess what they mean, and that's the point. The element <hgroup> however, may not be as obvious as the others. This element simply defines a group of header elements like <h1> to <h6>. Think of it as the title and subtitle of the articles.

4.1.1 The HTML

To show this we are going to use the most common layout there is: The 2-column layout. Normally this is done with a extensive use of <div> or <table> elements. In HTML5 this can be much more elegant.

```
<meta http-equiv="Content-Type" content="text/html; charset=utf-8">

<html>
    <head>
        <title><!-- Your Title --></title>
    </head>

    <body>
        <header>
            <!-- ... -->
        </header>

        <nav>
            <!-- ... -->
        </nav>

        <article>
            <hgroup>
                <h2>Title</h2>
                <h3>Subtitle</h3>
            </hgroup>

            <p>
                <!-- --->
            </p>
        </article>

        <footer>
            <!-- ... -->
        </footer>
    </body>
</html>
```

4.1.2 The CSS

In HTML4 and XHTML the code would normally be a <div> element. But with HTML5 we can use the names instead.

```
header, nav, article, footer, address {
    display: block;
}
```

Just to be on the safe side we declare all elements.

4.1.3 The problem

Let us check the design in some browsers. In Chrome and Firefox the page is being displayed like it should, and you have 4 distinguishable blocks, however in IE6 to IE8 the design isn't showing as it should - the styling isn't being applied to the page. By explicitly setting display: block; in CSS, we should have communicated to the browser our intentions for that element. Unfortunately, IE is ignoring elements it doesn't recognize, regardless of CSS. Our content is left floating in its parent's container, as if the HTML5 elements didn't exist. Somehow, we need to get IE to render unknown elements, and styling them appropriately isn't going to do it.

4.1.4 The solution

By the use of javascripting we can force IE to recognize the elements it ignored. The method was first used by John Resig [10], who calls it "HTML5 shiv". Simply put, it involves calling document.createElement() for every new unrecognized element. Traditionally you'd make this call in order to inject an element directly into some branch of the DOM; in other words, into an existing container within the <body> tag. You can do that to fix this unknown element issue as well. However, this trick also works by calling document.createElement() in the <head> tag, with no refence to a containing element. That makes it much easier to read and write:

```
document.createElement("footer");
document.createElement("header");
document.createElement("hgroup");
document.createElement("nav");
```

Lets try again with the javascript. With the added javascript, HTML5 now renders in IE6 and IE8 without any issues. This means that the developer by applying some lines of code, can make new HTML standards work in older browsers. HTML5 is exciting for the webdesigner and by using a few hacks it is posible to unleash the functionality of it in the browsers in use today.

Chapter 5

Conclusion

In the following chapter we will attempt to evaluate on the information we have presented through this report.

5.1 Using standards-only HTML

Given that all developers (including all current and future web-sites) conformed to standards, the initial workload would increase dramatically. A lot of time (and thus money) would have to be spent educating developers and having them update existing sites. The long-term effects of such a change are harder to determine. On one hand, developers would not have to create modifications for each browser. On the other hand however, more time would have to be spent to ensure that content is standards-compliant. On the browser-side of such a change, browser-developers would no longer be obligated to implement interpretations of non-standard HTML. This could free up more time for security-updates and implementations of new functionalities. The vast majority of browser-updates does however not relate to the interpretation of HTML and therefore we do not believe it would introduce a massive change in terms of browser-development.

5.2 Metadata Analysis and Mining Application (MAMA)

From the MAMA report (section 1.3.3) we see that website developers in general do not produce standards compliant code. Even pages produced by widely used CMS systems do not ensure compliant code. This gives us reason to believe that there is a general mindset amongst developers to disregard standards. We have not been able to factually determine the reason for this, the MAMA report does however suggest some possible explanations: [9]

- Many sites are built upon CMSes that do not spit out standards-compliant markup on the front end - it is nigh-on impossible to get these sites to validate.

- Many sites are put up on the Web by hobbyists, who do not care about Web standards - they just want to get their ”look at my kittens” site on the Web by any means necessary.
- Many sites these days feature user-generated content (think of any blog and social networking site); even if you make your blog validate, it can still easily be invalidated by a site visitor submitting a comment featuring bad markup.
- A lot of developers don’t care about validation - their site works for the target audience they are aiming it at, and they get paid regardless of standards compliance.
- And many more reasons!

5.3 Browser updating

Through our examination of the updates of our three major browsers we could determine that only a few of the updates have an actual impact on the interpretation of HTML. Only the major browser updates have had this kind of an impact. Because the major updates sometimes either stop supporting a hack or some non standard code, it is recommended for the developer to use the W3C standards, to keep the code working in the future versions of the browsers. Much of the old content on the internet still works because it was coded using the standards. Working with HTML5 and CSS3 taught us that even with the new standards, there shouldn’t be any problems with old websites, as the new versions only add new features, while having no effect on the current websites. They will be interpreted according to the legacy standards. This means valid pages still will be valid, and that badly coded pages still might work just as well or bad as they do today.

Bibliography

[1] http://www.w3.org/history/1989/proposal.html, 1989.
The original proposal of the WWW, HTMLized
The World Wide Web Consortium (W3C) is an international community where Member organizations, and the public work together to develop Web standards.

[2] http://www.w3.org/markup/sgml/, 1995.
Overview of SGML Resources.

[3] http://www.w3.org/tr/1998/rec-html40-19980424/, 1998.
HTML 4.0 Specification.

[4] http://www.w3.org/style/liebos2e/history/, 2001.
Cascading Style Sheets, designing for the Web - Chapter 20: The CSS saga.

[5] http://www.w3.org/tr/xhtml1/, 2002.
XHTML 1.0: The Extensible HyperText Markup Language (Second Edition).

[6] Why web standards matter.pdf, 2002.
http://netconnect.com - Team Portal.

[7] http://virtuelvis.com/archives/2005/01/css-history, 2005.
Who created CSS? CSS Early History
Arve Bersvendsen is a developer for Opera Software ASA.

[8] http://dev.opera.com/articles/view/mama-key-findings, 2008.
MAMA: Key findings - Dev.Opera.

[9] http://dev.opera.com/articles/view/mama-markup-validation-report/, 2008.
MAMA: Markup validation report - Dev.Opera.

[10] http://ejohn.org/blog/html5-shiv/, 2008.
John Resigs blog, HTML5 Shiv.

[11] http://www.mozilla.org/about/, 2008.
About Mozilla - official website
We find Mozilla as a reliable source.

[12] http://www.mozilla.org/about/history.html, 2008.
History of the Mozilla Project.

[13] http://www.mozilla.org/about/mission.html, 2008.
The Mozilla Misson.

[14] http://sixrevisions.com/web-development/the-history-of-web-browsers/, 2009.
The History of Web Browsers - Six Revisions
Articles are written by professionals.

[15] http://www.w3.org/tr/css3-selectors/, December 2009.
W3C's description of CSS3 Selectors.

[16] Emerging web standards.pdf, 2010.
http://Econtentmag.com - Digital Content Strategies & Resources.

[17] http://hsivonen.iki.fi/doctype/, 2010.
Activating Browser Modes with Doctype
Henri Sivonen, Master of Science (Technology), Thesis was about HTML5.

[18] http://www.instantshift.com/2010/10/15/the-history-of-web-browsers/, 2010.
The History of Web Browsers
Written by Daniel Pintille, working for Go-gulf.com, based in Dubai.

[19] http://www.w3.org/standards/webdesign/htmlcss, 2010.
HTML & CSS - W3C.

[20] http://www.w3resource.com/css/assigning-property-values-cascading-inheritance/css-origin-and-order-of-cascading.php, 2010.
Origin of Cascading - W3Resources
Founded in October 2008, w3resource.com was created as a comprehensive learning resource and experienced web developers can use it a reference.

[21] http://www.w3schools.com/css/css_intro.asp, 2010.
CSS Introduction - W3Schools.

[22] http://www.yourhtmlsource.com/accessibility/xhtmlexplained.html, 2010.
XHTML Explained - XHTML tutorial with code examples
Ross Shannon, Ph.d Computer Science.

[23] http://www.yourhtmlsource.com/starthere/historyofhtml.html, 2010.
The History of HTML - From the HTML 1.0 spec to XHTML 1.0...

[24] http://www.youtube.com/watch?v=jgmo7oximw8, 2010.
The story behind Google Chrome, by Google
This video was uploaded by Google.

[25] http://html5test.com/results.html, 2011.
HTML 5 test results.

[26] http://w3schools.com/cssref/css3_browsersupport.asp, 2011.
W3Schools Browser Support Reference.

[27] http://w3schools.com/cssref/css_selectors.asp, 2011.
W3Schools CSS Selector Reference.

[28] http://www.w3.org/tr/html5-diff/, May 2011.
W3C's description of HTML5 differences from HTML4.

[29] http://www.w3schools.com/browsers/browsers_stats.asp, 2011.
Browser Statistics
These are from W3shools logs, and are being revised.