Tolkningsskillnader mellan webbläsare

En studie om faktorer som förorsakar tolkningsskillnader mellan webbläsare

Jan Nyström

Examensarbete

Mediekultur

EXAMENSARBETE	
Arcada	
Utbildningsprogram:	Mediekultur
Identifikationsnummer:	2970
Författare:	Jan Nyström
Arbetets namn:	**Tolkningsskillnader mellan webbläsare** - En studie om faktorer som förorsakar tolknings-skillnader mellan webbläsare
Handledare (Arcada):	
Uppdragsgivare:	

Sammandrag:

Denna studie går ut på att forska i vilka faktorer som förorsakar skillnader mellan hur olika webbläsare presenterar nätsidor och hur man kan minimera dessa skillnader redan i planeringsskedet av nätsidan för att undvika eventuella tidskrävande korrigeringar i efterhand.

Det centrala problemet som behandlas i den här forskningen är att en stor del av tiden då man skapar webbplatser endast går till att försöka lösa hur man skall få nätsidorna att fungera som den skall i olika webbläsare.

Syftet med forskningen är att lära sig förstå vad som förorsakar skillnader mellan webb-läsare, för att effektivare kunna skapa nätsidor med bra funktionalitet mellan olika webbläsare och dessutom spara tid.

Själva forskningen går ut på att gå igenom hur webbläsare tolkar nätsidor, vilka faktorer som förorsakar tolkningsskillnader, hur webbläsarstatistiken ser ut idag, vilka gemen-samma egenskaper dessa webbläsare som används idag har och hurdana metoder man kan använda sig av för att skapa nätsidor med god funktionalitet mellan webbläsare.

Genom att känna igen vad som förorsakar tolkningsskillnader mellan webbläsare och planera nätsidan så att fonter och formulär har rum att ”leva” mellan olika webbläsare kan man både spara arbetstid och få nätsidan att fungera bättre i olika webbläsare.

Nyckelord:	webbläsare, tolkningsskillnader, cross-browser
Sidantal:	45
Språk:	Svenska
Datum för godkännande:	

DEGREE THESIS	
Arcada	
Degree Programme:	Mediekultur
Identification number:	2970
Author:	Jan Nyström
Title:	**Rendering differences between web browsers** - A study in factors that causes web browsers to render web pages differently
Supervisor (Arcada):	
Commissioned by:	

Abstract:

This study is about doing a research in what factors causes differences between how web browsers presents web pages and how to minimize these differences in the planning stage of a web page to avoid possible time consuming corrections later on.

The main problem that this research focuses on is that a lot of the time when producing a website is spent only on trying to solve how to get web pages to work like they should in different web browsers.

The goal with this research is learning to understand what causes differences between web browsers so that one could create web pages more efficiently with good functionality between web browsers and also spare working time in the process.

The research itself is about going trough how web browsers interpret web pages, which factors causes interpretation differences, how the web browser statics looks like today, which common abilities the web browsers have that are used today and what kind of methods can be used to create web pages with good functionality between web browsers.

By knowing what causes the web browsers to display web pages differently and by planning the web pages in a way that let text and forms have a little room to "live" between different web browsers, it's possible to spare time and make the web page perform better in different web browsers.

Keywords:	
Number of pages:	45
Language:	Swedish
Date of acceptance:	

OPINNÄYTE	
Arcada	
Koulutusohjelma:	Mediekultur
Tunnistenumero:	2970
Tekijä:	Jan Nyström
Työn nimi:	**Verkkoselainten välisiä esityseroja** - Tutkimus tekijöistä jotka aiheuttaa verkkoselainten välisiä esityseroja
Työn ohjaaja (Arcada):	
Toimeksiantaja:	

Tiivistelmä:

Tutkimuksen tavoitteena on selvittää mitkä tekijät aiheuttavat eroja siinä miten eri verkkoselaimet esittävät verkkosivuja ja miten näitä esityseroja voidaan minimoida jo suunnitteluvaiheessa välttääkseen mahdollisia jälkikäteen tehtäviä aikaanvieviä korjauksia.

Keskeisin ongelma jota tässä tutkimuksessa käsitellään, on se miten paljon aikaa verkkosivujen luomisessa kuluu pelkästään siihen että yritetään ratkaista miten ne saataisiin toimimaan niin kuin niitten kuuluu eri verkkoselaimissa.

Tutkimuksen tavoitteena on oppia ymmärtämään, mikä aiheuttaa verkkoselainten välisiä esityseroja ja miten pystytään tehokkaammin luomaan verkkosivuja jotka toimivat hyvin eri verkkoselaimissa ja näin myös säästää työaikaa.

Itse tutkimuksessa käydään läpi miten verkkoselaimet esittävät verkkosivuja, mitkä asiat vaikuttavat esityseroihin, miltä verkkoselainten tilastot näyttävät tänään, millaisia yhtenäisiä ominaisuuksia niillä verkkoselaimilla on joita tänä päivänä käytetään ja millaisia menetelmiä voidaan käyttää luodakseen verkkosivuja joilla on hyvä toimivuus eri verkkoselaimissa.

Tiedostamalla mitkä asiat saavat verkkoselaimet näyttämään verkkosivuja eri tavalla ja suunnittelemalla verkkosivuja niin että tekstille ja lomakkeille annetaan tilaa ”elää” eri verkkoselainten välillä, on mahdollista säästää työaikaa ja sammalla saada verkkosivun toimimaan paremmin eri verkkoselaimissa.

Avainsanat:	
Sivumäärä:	45
Kieli:	Ruotsi
Hyväksymispäivämäärä:	

INNEHÅLL / CONTENTS

Figurer

Tabeller

1 INLEDNING

1.1 Ämne och motiv för ämnesvalet

När jag skapat en nätsida märker jag ofta att olika webbläsare har en del variationer på hur de presenterar nätsidan. Ibland kan det vara mindre detaljer som t.ex. att olika webbläsare presenterar formulärfält på olika sätt eller att de använder sig av olika fonter. Då variationerna mellan olika webbläsare är så stora att sidans struktur bryts ut och gör det svårt för besökaren att läsa informationen är det någonting man ofta bör korrigera så fort som möjligt för att inte ge ett dåligt intryck åt det antal besökare som använder sig av en annan webbläsare. Ofta märker jag att en nätsida som i början varit menad att vara ett kort projekt kan bli väldigt långt på grund av alla korrigeringar och kompromisser som i efterhand måste göras för att nätsidan skall fungera och se bra ut i en mängd andra webbläsare.

Jag har fått en sådan bild att det skulle finnas stort behov av nätsidor för mindre företag, föreningar och enskilda personer men att det är få företag, personer som vill ta emot sådana mindre projekt. En bekant till mig vars arbete bl.a. är att rekrytera kodare för nätsidor ansåg det inte vara lönsamt att ta emot projekt som ger under 3000~4000 euro.

Det jag anser om projekt för nätsidor där budgeten endast är några hundra euro, är att det inte är värt mödan som krävs för att få nätsidan att fungera i olika webbläsare, i synnerhet då man inte är insatt i ämnet om vad som förorsakar tolkningsskillnader mellan webbläsare. Det kan vara svårt att övertyga en kund att betala mera bara för att det är så tidskrävande att få en nätsida att fungera i olika webbläsare. Att hålla en diskussion med en kund på en professionell nivå och samtidigt förklara om tolkningsskillnader är inte en bra lösning till problemet.

I mitt examensarbete kommer jag att forska i vilka faktorer som förorsakar skillnader mellan hur olika webbläsare presenterar nätsidor och hur man kan minimera dessa skillnader under utvecklingsstadiet av nätsidan för att minimera eventuella tidskrävande korrigeringar i efterhand.

1.2 Syfte

Syftet med mitt examensarbete är att ge mig själv och läsaren en bättre insikt i vilka faktorer som förorsakar tolkningsskillnader mellan webbläsare. Min tanke är att då man är medveten om dessa faktorer, kan man betydligt minska på eventuella skillnader mellan webbläsare och på så sätt både spara tid och ge kunden ett professionellt intryck.

1.3 Frågeställningar

1. Kan man med dagens webbtekniker skapa nätsidor som är pixelnoggranna mellan moderna webbläsare?
D.v.s. om t.ex. en grafiker skapar en layout för en nätsida, kan man då göra om designen till en nätsida så att om man tar en skärmbild av nätsidan och jämför den med den ursprungliga designen inte hittar någonting avvikande?

2. Vilka faktorer inverkar på hur olika webbläsare tolkar nätsidor?
D.v.s. vad är det som inverkar på hur olika webbläsare presenterar nätsidor och vad innebär det?

3. Hur kan man minska på tolkningsskillnader mellan webbläsare?
D.v.s. hur kan man med olika metoder minska eller undvika tolkningsskillnader mellan webbläsare.

1.4 Materialbeskrivning

Mitt material består till största del internetkällor. Orsaken till detta är att utvecklingen inom internet tekniker går så snabbt framåt att det som en författare ansett för två år sedan inte behöver stämma. Då jag samlat in material för fakta använder jag litteratur för att bekräfta att informationen är legitim men använder även internetkällor som stöd för formuleringen.

För att samla in statistik om webbläsare använder jag mig av två olika internet källor för att ge en bättre helhetsbild om vilka webbläsare som används idag. Till mitt material hör även en del egna experiment.

Jag gör egna experiment och jämförelser för att lättare kunna demonstrera och bekräfta skillnader i hur webbläsare tolkar nätsidor.

1.5 Metodredovisning

Min forskningsmetod går ut på att först gå igenom hur webbteknikerna utvecklats och hur de fungerar. Då webbteknikerna presenterats gör jag en forskning i vad som förorsakar skillnader mellan webbläsare och försöker även finna svaren till varför. Då jag behandlat webbläsarskillnaderna presenteras metoder med vilka man kan minska på tolkningsskillnader och till sist summerar jag ihop det jag lärt mig.

Jag kommer att använda mig av olika experiment för att demonstrera skillnader mellan hur webbläsare tolkar nätsidor. För att komma på rätt spår om hurdant material jag skall använda mig av, kommer jag att använda mig av internet för leta efter artiklar och diskussioner om tolkningsskillnader mellan webbläsare. Eftersom Internets utveckling går så snabbt framåt vill jag komma fram med sådant material som är aktuellt idag. Sådant som tidigare ansets vara bra metoder kan idag anses vara dåliga. Därför vill jag förhålla mig kritiskt till såväl påståenden som nämns i böcker och på internet såväl som till mina egna kunskaper.

1.6 Målgrupp

Det här arbetet är i första hand riktat till webbdesigners med förkunskaper i HTML och CSS men även grafiker som är intresserade av att designa layouts för nätsidor kan ha nytta av den här forskningen.

1.7 Avgränsningar

JavaScript

Jag har valt att inte gå in på hur man med JavaScript kan skapa webbläsarspecifika lösningar för bättre funktionalitet mellan olika webbläsare. Orsaken till detta är att webbläsaren måste stöda JavaScript och att man ibland även med JavaScript måste göra webbläsarspecifika definitioner. I denna forskning då ett av målen är att man skall spara tid vill jag hålla mig till så simpel teknik som möjligt.

Flash

Flash används mycket på nätet men eftersom Flash är en utomstående applikation så har inte webbläsarskillnaderna någonting att göra med hur själva Flash innehållet presenteras. Ett problem med Flash sidor är att de lätt blir mera komplexa att skapa och att man måste hitta på ett sätt att även berätta för sökmotorer som bara förstår sig på HTML hur sidan är strukturerad och vad den innehåller ifall man vill få besökare till sidan.

HTML5

Jag tänkte först ta med HTML 5 men ansåg det inte vara relevant för det här arbetet. HTML 5 är inte ännu en W3C rekommendation och dess mest intressanta nya egenskaper ligger i multimedia egenskaper.

2 HTML

Förkortningen HTML kommer från ***H****yper* ***T****ext* ***M****arkup* ***L****anguage. HTML* är ett märkspråk, HTML-dokumentet innehåller särskilda element som inte visas då dokumentet presenteras i sin slutliga form utan ger endast direktiv för webbläsaren om hur dokumentet skall struktureras. I HTML-dokumentet kallas dessa element för tags. Dessa definieras med ett större än och mindre än tecken. Nedanför ett exempel på ett simpelt HTML-dokument. (McFarland 2009:3)

Ett mycket simpelt HTML-dokument:

```
<html>
<head>
<title> Nätsidans rubrik </title>
</head>
<body>
<h1> Huvud rubrik </h1>
<p> En paragraf som innehåller text, <b> Fet stil </b>, <i> Kursiv stil </i> </p>
</body>
</html>
```

HTML dokumentet presenterat i en webbläsare:

Figur 1. Ett simpelt HTML-dokument presenterat i en webbläsare.

2.1 Utvecklingen

När internet först skapades av en grupp av vetenskapsmän som ett hjälpmedel för att dela med sig och hålla kontroll på olika tekniska dokumentationer, var inte presentationsstilen det man i första hand ansåg vara viktigt. HTML gjorde precis det som vetenskapsmännen behövde, nämligen att strukturera informationen för att lätt kunna förstås. (McFarland 2009:18).

2.1.1 Formgivning

Då andra förutom vetenskapsmän började använda sig av HTML ville de få sina nätsidor att se bra ut. Man började använda sig av tag-elementen för att kontrollera utseende istället för att bara strukturera informationen. Ursprungligen var inte tag-elementen i HTML avsedda för formgivning av dokumentet utan enbart för att definiera innehållet i dokumentet. (McFarland 2009:19)

När man i HTML versionen 3.2 lade till tag-element som <font> förorsakade det en massa problem för webbdesigners. För stora webbplatser innebar det att man hamnade lägga in typsnitt och färg definitioner i varenda ny nätsida och för varje enskilt element som innebar en massa extra arbete, dessutom blev sidorna även stora och tunga att ladda. (McFarland 2009:19), (w3schools.com - Why use HTML 4.0?), (w3schools - CSS)

I följande exempel kan man se hur ett HTML-dokumentet presenteras i en webbläsare då man lägger till formgivning.

Ett HTML dokument som använder sig av formgivning i <font> elementet:

```
<html>
<head>
<title> Nätsidans rubrik </title>
</head>
<body>
<h1> <font face="arial" size="6"  color="red"> Huvud rubrik  </font> </h1>
<p><font face="verdana" size="3"  color="green">Det här är själva texten!</font></p>
</body>
</html>
```

HTML-dokumentet presenterat i en webbläsare:

Figur 2. Ett exempel på hur inbakade presentationsstilar presenteras i en webbläsare.

2.1.2 Presentationsstil

HTML 4.0 var den första versionen av HTML där man kunde lämna bort all formgivning från HTML-dokumentet och spara presentationsstilen i en extern stilmall som kallas för CSS ”***C**ascading **S**tyle **S**heets*”, behandlas i kapitel 3. Detta gav webbdesignare full kontroll över presentationsstilen utan att behöva stöka till HTML-dokumentet.

Tack vare den externa presentationsstilen kunde man nu länka flera nätsidor till samma presentationsstil och behövde inte längre skriva om presentationsdefinitioner i alla HTML-dokument varje gång man ville uppdatera presentationsstilen.
(w3schools.com - Why use HTML 4.0?), (McFarland 2009:19)

Från och med HTML 4.0 kan man använda sig av externa presentationsstilar:

```
<head>
<link rel="stylesheet" type="text/css" href="presentationsstil.css">
</head>
```

2.2 Dokumenttyper

En dokumenttyp deklaration ”*doctype declaration*” hänvisar till en DTD “*Document Type Definition*” som specificerar reglerna för märkspråket så att webbläsaren kan presentera innehållet korrekt. Dokumenttyp deklarationen är inte en HTML tag utan en instruktion för webbläsaren om vilken version av ett märkspråk sidan är skriven med.

Om man skriver dokumenttyp deklarationen fel kan man få webbläsaren att ändra sitt tillstånd till ett såkallat ”quirks mode” tillstånd. Quirks mode är webbläsartillverkarnas försök att få webbläsaren att fungera som webbläsare fungerade ungefär omkring 1999. Det kan innebära att webbläsaren tror att nätsidan är skriven för länge sedan och försöker presentera sidan så som den skulle presenterats med en föråldrad webbläsare.
(McFarland 2009:26), (Murphy.Persson 2009:28),
(w3schools.com - HTML <!DOCTYPE> Declaration)

2.2.1 Strict

Innehåller alla HTML element och attribut, men innehåller inte presentations attribut eller ogillade element som t.ex. <font>. Användningen av ”framesets” är förbjudet.
(w3schools.com - CSS Id and Class)

2.2.2 Transitional

Innehåller alla HTML element och attribut, tillåter även presentations attribut samt ogillade element som t.ex. <font>. Användningen av ”framesets” är förbjudet.
(w3schools.com - CSS Id and Class)

2.2.3 Frameset

Innehåller samma regler som Transitional men tillåter även användningen av framesets.
(w3schools.com - CSS Id and Class)

2.3 XHTML

XHTML står för ”*e**X**tensible **H**yper**T**ext **M**arkup **L**anguage”* och är en kombination mellan HTML 4.01 samt XML ”*e**X**tensible **M**arkup **L**anguage*”. XHTML består av alla element som finns i HTML 4.01 kombinerat med den strikta syntaxen från XML ”e**X**tensible **M**arkup **L**anguage”.

De viktigaste skillnaderna mellan XHTML och HTML

- XHTML kräver att man alltid börjar dokumentet med en dokumenttyp.
- XHTML elementen måste alltid avslutas.
- Element som är insatta i varandra måste avslutas i rätt ordning och alla element måste skrivas med små bokstäver.

Följande märkspråk bryter mot reglerna i XHTML, elementen är avslutade i fel ordning, stora bokstäver används i elementen och ”*line break ”*
” elementet har inte avslutats.¨

```
<P><b>TEXT </p></B>
<br>
```

Nedanför en korrigerad version.

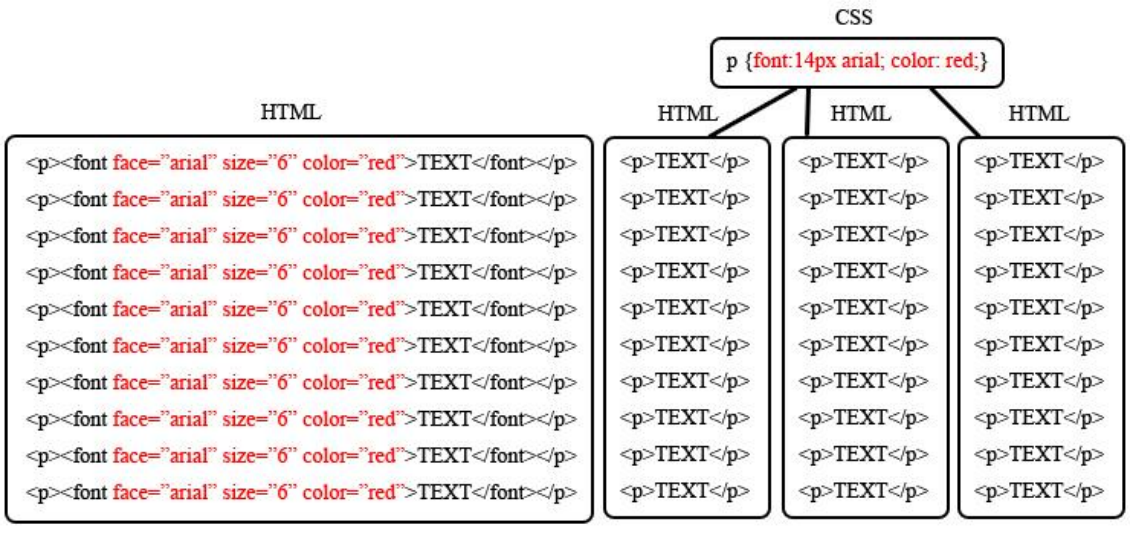
```
<p><b> TEXT </b></p>
<br />
```

Eftersom XHTML har striktare regler om hur dokumentet bör formuleras jämfört med HTML är det i synnerhet webbläsare för mobila apparater som har nytta av XHTML eftersom de inte har de resurser eller kraft att tolka ett uselt skrivet märkspråk.
(McFarland 2009:5-6), (w3schools.com - XML Tutorial),
(w3schools.com - XHTML Tutorial)

3 CSS

CSS står för ”***C**ascading **S**tyle **S**heets*” och är en stilmall som innehåller stilar för att definiera hur HTML-element skall presenteras. Genom att använda sig av stilar istället för stilattribut i HTML-elementen kan man spara en massa arbete såväl då man skapar en webbplats som då man gör små uppdateringar. Eftersom man lagrar stilarna i ett skilt CSS-dokument som man kan länka till flera olika HTML-dokument, sparar man ännu mera arbete eftersom webbplatser ofta har en mängd HTML-dokument för de olika nätsidorna men ett gemensamt dokument med presentationsstilen.
(w3schools - CSS Introduction)

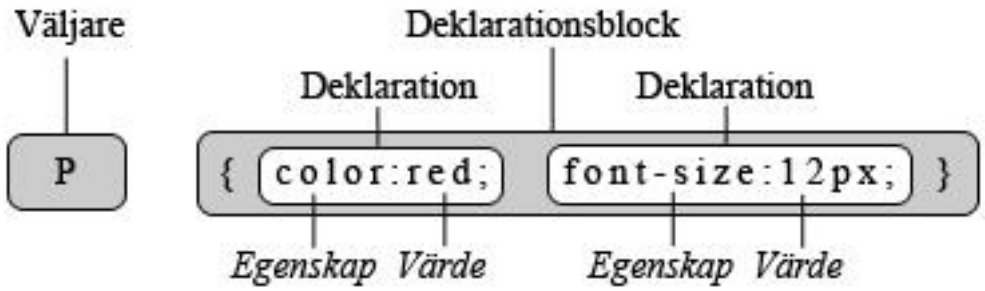
I exemplet nedanför kan man se hur mycket arbete man kan spara genom att använda sig av CSS. Till vänster ser man först ett HTML-dokument där varje paragraf-element har stildefinitioner i HTML-elementet. Till höger har vi tre HTML-dokument som använder sig av samma CSS-dokument. Om man vill ändra textens färg i HTML-dokumentet till vänster måste vi ersätta ”red” med t.ex. ”blue” tio gånger medan man för de tre HTML-dokumenten till höger behöver göra detta en gång i CSS dokumentet.



Figur 3. Presentationsdefinitionerna är färgade med rött för att lättare urskiljas.

3.1 Stilar

En stil är uppbyggd av en väljare ”selector” och ett deklarationsblock ”declaration block”. Väljaren kan t.ex. vara en huvudrubrik, en paragraf, en bild osv. I deklarationsblocket kan man definiera hur man vill att elementet man valt skall presenteras. Man kan t.ex. ge paragrafer färg, bilder ramar eller ange positioner och marginaler. Det finns oändligt med möjligheter. Följande exempel demonstrerar hur en stillmall är strukturerad. (w3schools - CSS Introduction), (Wikipedia.org - Cascading Style Sheets), (McFarland 2009:31-33)



Figur 4. En illustration om hur en stilmall är strukturerad.

3.2 Väljare

Med CSS kan man förutom att definiera stilar för HTML-element även specificera egna väljare som kallas för Id och Class. (w3schools.com - CSS Id and Class)

3.2.1 Id

Id-väljaren används för att specificera en stil för ett ensamt, unikt element. I HTML-dokumentet används attributet Id för att definiera elementet och i CSS definieras Id med ett nummertecken. (w3schools.com - CSS Id and Class)

HTML:

```
<p id=”uniktext”> Brödtext </p>
```

CSS:

```
#uniktext {color:red;}
```

3.2.2 Class

Class väljaren används för att specificera stilar för en grupp element. Till skillnad från Id-väljaren används Class väljaren ofta för ett antal element. Detta tillåter en att definiera en specifik stil för vilket HTML element som helst vars Class-attribut är det samma. I CSS definieras Class-väljaren med ett punkt tecken.
(w3schools.com - CSS Id and Class)

HTML:

```
<h1 class=”gemensamfarg”>En rubrik som blir röd</h1>
<p class=”gemensamfarg”>Brödtext som blir röd</p>
```

CSS:

```
.gemensamfarg { color:red;}
```

3.2.3 Avancerad väljning

Med ett kommatecken kan man separera flera väljare för samma deklarationsblock.

HTML:

```
<h1>RUBRIK</p>
<h2>UNDERRUBRIK</p>
<p>CITAT</p>
```

CSS:

```
h1,h2,p { color:red;}
```

Man kan specifikt välja endast sådana element som befinner sig innanför ett annat element genom att specificera flera element i samma väljare och skilja åt dem med ett mellanslag.

HTML:

```
<h2>DEN HÄR RUBRIKEN BERÖRS INTE</h2>
<div class=”kontainer”>
<h2>DEN HÄR RUBRIKEN BLIR RÖD</h2>
</div>
```

CSS:

```
.kontainer h2 { color:red;}
```

(w3schools.com - CSS Grouping and Nesting Selectors)

4 SKILLNADER MELLAN WEBBLÄSARE

För att man bättre skall kunna förstå vad som förorsakar skillnader mellan hur webbläsare visar nätsidor är det viktigt att man förstå hur nätsidorna är uppbyggda. En nätsida är uppbyggd av ett märkspråk, vanligen HTML *Hyper Text Markup Language*, ett strukturerat dokument som innehåller strukturerat innehåll och av en stilmall som kallas för CSS ”*Cascading Style Sheets*”, som antingen är länkad till HTML-dokumentet eller inbakad i koden och vars uppgift är att beskriva presentationsstilen för HTML-dokumentet. Så här beskriver McFarland om hur dessa bör användas:

> “Use HTML to organize your content and CSS to make that content look great” (McFarland 2009:19)

För att kunna läsa HTML-dokumentet behövs en webbläsare. Olika webbläsare tyder nätsidorna på olika sätt. Man kan till exempel jämföra webbläsare med tolkar. Deras uppgift är att översätta någonting från ett främmande språk till ett språk som vi förstår. Om man ger flera tolkar ett dokument som är skrivet på ett främmande språk så kan tolkarna översätta dokumentet så att själva budskapet kommer att vara det samma men deras översättningar kommer att variera. (netmechanic.com - Browser Compatibility Tutorial)

Då man med en webbläsare besöker en nätsida ser den HTML-dokumentet som innehåller strukturerad text och vanligtvis en stillmall. Det är webbläsaren uppgift att på basen av HTML-dokumentet och den angivna stilmallen skapa en presentationsstil för användaren. Webbläsaren skapar presentationsstilen i första hand på basen av stillmallen. I fall det i HTML-dokumentet finns ett element som inte definieras exakt i stilfilen använder sig webbläsaren av sina egna färdigt inställda värden för att skapa presentationsstilen. (McFarland 2009:102)

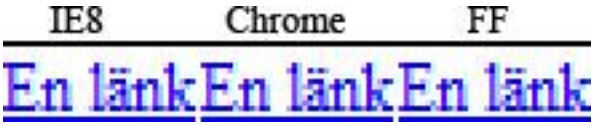
4.1 Olika tolkningssätt

I Figur 5 nedan kan man se tre olika variationer av hur en länk presenteras i olika webbläsare då HTML-dokumentet inte använder sig av någon presentationsstil för länken. Alla tre webbläsare ger olika variationer på länk-elementet eftersom de använder sig av egna inbyggda stil dokument. Längst till vänster har vi IE 8 som ger länken en aning ljusare färg än de andra två webbläsarna. I mitten har vi webbläsaren Chrome som presenterar länken med samma höjd som IE8 men understräckningen har bara ett pixels mellanrum från texten medan de andra två webbläsarna använder sig av två pixels mellanrum. Till höger har vi Firefox vars text ligger en pixel lägre ned än hos de andra två webbläsarna.

Figur 5. Tolkningsskillnader mellan IE8,Chrome och Firefox.

4.2 Gemensam presentationsstil

Figur 5 ger en bra översikt om hur olika webbläsare egentligen tolkar nätsidor. De att någonting ser bra ut med en webbläsare betyder absolut inte att den då ser lika ut i alla andra webbläsare. Genom att lägga in en mängd beskrivningar i stilmallen får vi dessa tre webbläsare att visa länken exakt på samma sätt. Källkoden för dokumentet finns med som bilaga på sidan 41.



Figur 6. Med stilar kan man få länkarna att se lika ut i olika webbläsare.

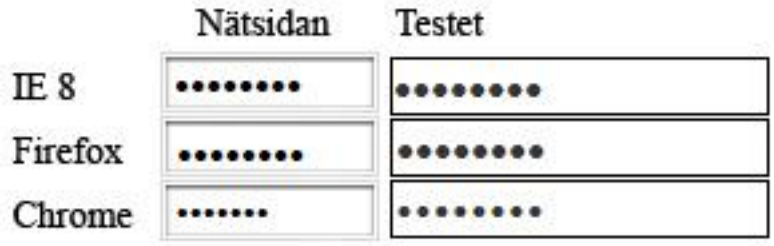
4.3 Formulär

Allting är inte lika lätt att få att se exakt lika ut i olika webbläsare. Formulärfält är så gått som omöjliga att fullständigt ge stil åt. (meyerweb.com – formal weirdness)

Man kan ibland stöta på små detaljer i formulär som att texten man skriver in i fältet inte befinner sig vertikalt exakt i mitten t.ex. då man skall logga in någonstans.
I följande exempel togs skärmbilder från en anonym nätsidas formulärfält med tre olika webbläsare. Webbläsarna som användes var IE8 Firefox och Chrome. Genom att med ett bildhanteringsprogram jämföra formulärfältet, kunde man notera följande skillnader mellan hur de olika webbläsarna presenterade formulärfältet.

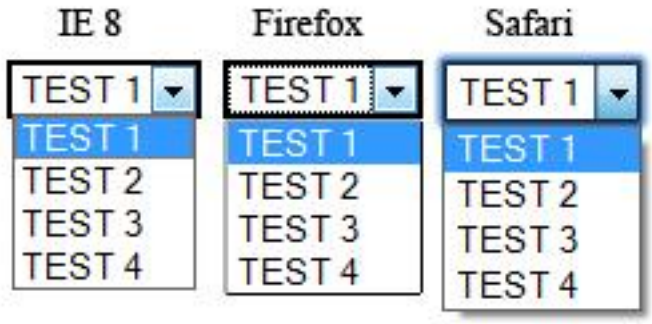
I figur 7 finns det två olika versioner av ett formulärfält för att ange ett lösenord. På vänstra sidan har tre skärmbilder av fältet tagits från olika webbläsare och radats på varandra för att gemföra skillnader. Formulärfälten till höger är ett försök att med CSS få fälten att se exakt lika ut i de olika webbläsarna. Källkoden för dokumentet som användes finns med som bilaga på sidan 42.

I den högra kolumnen kan man notera att IE8 visar cirklarna vertikalt i mitten medan Firefox visar dessa en aning lägre ned. Chrome visar cirklarna vertikalt i mitten men de är mycket mindre än cirklarna i IE8 och Firefox. I den högra kolumnen i figur 7 har ett test gjorts med CSS för att få ett formulärfält se exakt likadant ut i de olika webbläsarna. I testet framgick det att man med CSS kunde få cirklarna att positioneras vertikalt i mitten men marginalerna och storleksskillnaderna förblev olika.



Figur 7. Formulär kan vara besvärliga att få och se exakt likadana ut.

Ett större problem gällande formulär är den såkallade ”*select*” menyn. Man kan till en viss mån med CSS ge stil även till detta element men t.ex. pilarna med den ljusblåa bakgrunden i figur 8 kan man inte ändra på. Många webbsidor använder därför sig av andra webbtekniker som t.ex. JavaScript för att skapa bättre presentationsstil för sina formulär.



Figur 8 CSS ingen makt över Select elementet

4.4 Fonter

Med hjälp av CSS kan man definiera för webbläsaren vilken font man vill att webbläsaren skall använda sig av då den presenterar nätsidan. Fonterna presenteras inte alltid exakt på samma sätt, detta beror i huvudsak på att olika operativsystem har olika färdigt inbyggda fonter, men detta kan även bero på webbläsaren. Nuförtiden brukar en del operativsystem stöda uppmjukning av fonter för behagligare läsning. Detta förorsakar inte direkt stora skillnader men kan töja ut fonten en aning.

För följande två exempel har ett HTML-dokument med strikt definierade stilar använts som botten för att lättare kunna konstatera skillnader i hur fonter skiljer sig mellan webbläsare. Källkoden för dokumentet som använts i detta exempel finns som bilaga på sidan 44.

I det första exemplet kan man se hur olika fonter varierar mellan Webbläsare och operativsystem. I Figur 9 har 13 skärmbilder från 13 olika kombinationer av webbläsare och operativsystem lagts ihop i tre kolumner enligt operativsystem. Skärmbilderna har gjorts delvis genomskinliga. Ju skarpare fonten är, desto mindre är skillnaderna.

Figur 9. Skärmbilder från ("www. browsershots.org" Hämtade 23.5.2010)

Man kan tydligt se att det huruvida webbläsaren klarar av att visa en font beror mera på vilket operativsystem som använts än på vilken webbläsare man använder sig av. I de två första kolumner kan man se att det bara är MS Serif fonten som skiljer sig kraftigt mellan de olika webbläsarna. Skillnaden mellan Windows 7 och Windows XP är endast den att Windows 7 mjukar ut fonterna en aning och att IE webbläsarna visar de flesta fonter en pixel högre upp. Arial, Trebuchet MS, Times New Roman och Lucida Console är pixelnoggranna i de två första kolumnerna.

Om vi nu tar och ser på hur webbläsare i den sista kolumnen visar dessa fonter under olika Linux operativsystem så är skillnaderna stora, den största skillnaden är över 25 pixel och då är det bara frågan om ett ord med färre en tjugo bokstäver.

I det andra exemplet jämförs skillnaderna mellan Operativsystemet Windows 7 och MAC OS X med två webbläsare. I figur 10 har fyra skärmbilder med sex olika kombinationer mellan de två operativsystem och webbläsare lagts ihop. På samma sätt som i figur 9 är skärmbilderna delvis genomskinliga för att jämföra skillnader.

Figur 10. Fonterna skiljer sig en aning från varandra

I figur 10 kan man igen se att det finns skillnader. Slutligen kan man konstatera att det inte finns fonter som skulle vara exakt pixelnoggranna mellan olika webbläsare. Det lönar sig alltså då man gör en grafisk design av en nätsida att ta i beaktande hur fonter kan variera en aning mellan olika webbläsare, och ge dem rum att ”leva” i designen så att nätsidan ser bra ut även om fonten inte befinner sig på exakt rätt plats.

5 MINIMERA TOLKNINGSKILLNADER

5.1 Val av CSS

I dag anses CSS 2.1 vara en standard, detta innebär dock inte att man genom att använda stilar som är klassade till högst CSS 2.1 garanterat undviker tolkningsskillnader mellan webbläsare. För att få en bild om vilka webbläsare som i dag används kan man kolla på olika webbläsarstatistiker.

Webbstatisk kan variera kraftigt som tabellen nedanför visar. W3schools.com samlar in webbläsarstatistik från deras besökare, denna sida kan man anta att besöks i huvudsak av webbdesigners. Webbläsarstatistiken från netmarketshare.com samlar in statistik från en mängd olika källor. I det här exemplet visas de olika webbläsarversioner av Internet Explorer skilt eftersom deras egenskaper skiljer sig kraftigt från varandra och deras procentuella andelar är mycket höga.

Det man tydligt kan konstatera i tabellen nedan är att Internet Explorer 6 ännu är en mycket använd webbläsare fastän den snart är tio år gammal.

April 2010	IE8	IE7	IE6	Firefox	Chrome	Safari	Opera
w3schools.com	16.2%	9.3%	7.9%	46.4%	13.6%	3.7%	2.2%
netmarketshare.com	24,7%	12,5%	17,6%	24.6%	6.7%	4.7%	2.3%

Tabell 1. Statistik samlat från w3schools.com samt netmarketshare.com 16.5.2010.

Följande tabell är från nätsidan www.quirksmode.com, en nätsida där det samlas in information on webbläsarkompatibilitet. Från deras tabell kan man konstatera att IE 6 och IE 7 inte är fullt CSS 2 kompatibla, medan redan mycket gamla versioner av, Firefox, Chrome samt Safari stöder CSS 2 till fullo.

Figur 11. ("http://www.quirksmode.org/compatibility.html", Hämtat 23.5.2010)

McFarland anser följande om webbläsarnas CSS stöd:

> "CSS 2.1 isn't a radical change from version 2, and most web browsers have adapted to the new rules just fine, thank you. (A notable exception is Internet Explorer 6 for Windows—that's why you'll find helpful workarounds for dealing with browser differences sprinkled throughout this book. Thankfully, Internet Explorer 7 fixed most of the hairpulling bugs of its predecessor, and Internet Explorer 8 finally follows almost all CSS 2.1 rules correctly.)" (McFarland 2009:9)

På basen av webbläsarstatistik om vilka webbläsare som används idag samt vilka egenskaper dessa har, kan man konstatera att alla egenskaper i CSS 2.1 standarden, inte fullständigt stöds av webbläsarna. Det att CSS 2.1 inte stöds av alla webbläsare betyder inte att man bör undvika att använda sig av 2.1 versionen. Eftersom det närmast är IE6 som har svårigheter med CSS 2.1 standarden lönar det sig hellre att göra webbläsarspecifika definieringar än att använda sig av en äldre CSS version.
(netmarketshare.com), (w3schools.com - Browser Statistics), (quirksmode.org)

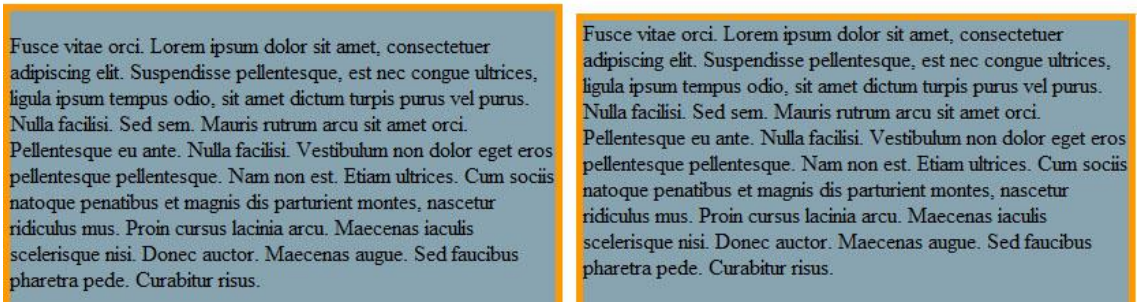
5.2 CSS-nollställning

Varje webbläsare har färdigt inställda värden de använder sig av för att presentera element i HTML dokumentet som inte specifikt angetts med CSS. Då man låter webbläsaren ge färdigt inställda värden för olika element är risken stor att det uppstår skillnader mellan hur olika webbläsare anser att sidan skall presenteras.

Jacob Gube har skrivit en artikel om CSS nollställning. Han har gjort en exempelsida för att demonstrera skillnaderna mellan webbläsare. I exemplet jämför Gube webbläsaren Mozilla Firefox 3 mot IE7. I en nätsida har paragrafer med text placerats in i ett element. Elementets uppgift är att fungera som en container för paragraferna.

Genom att öppna Gubes exempelsida med IE8 kan man notera att den presenterar sidan på samma sätt som Firefox 3.6.2. Eftersom IE8 har en inbyggd funktion som får den att fungera som IE7, kan man notera att det även fanns skillnader mellan hur de två olika versioner av IE presenterar sidan.

Poängen är dock att det inte spelar någon roll vilken av webbläsarna som har rätt eller fel. Problemet är det att sidans presentationsstil inte är gemensam i alla webbläsare. Eftersom man inte angett egna värden för paragraferna använder sig webbläsarna av egna färdigt inställda värde för paragrafen. Genom att i presentationsstilen ange vad marginalen för paragrafen bör vara, kunde man undvika dessa tolkningsskillnader. (sixrevisions.com - CSS Tip #1: Resetting Your Styles with CSS Reset)



Figur 12. (”http://sixrevisions.com/demo/reset_styles/example1.html” Hämtat 14.5.2010)

5.2.1 Olika nollställningar

CSS-nollställning handlar i praktiken om att man i CSS-dokumentet nollställer alla värden för alla element i HTML-dokumentet så att man får en gemensam startpunkt för olika webbläsare och såvida minskar på tolkningsskillnaderna.

Det finns inte någon exakt regel på hur man bör nollställa CSS stilen, det beror också på vilka element man använder sig av i HTML-dokumentet. Man kan fråga sig att är det verkligen nödvändigt att nollställa sådana element som man inte kommer att använda eller om man vet att man ändå kommer att definiera elementet på nytt.

Det finns massor av olika exempel på nätet om hur man kan nollställa presentationsstilen med CSS.

Från följande webbadress (”*http://perishablepress.com/press/2007/10/23/a-killer-collection-of-global-css-reset-styles/, Hämtat 23.5.2010)* kan man hitta artikeln ”*A Killer Collection of Global CSS Reset Styles”* där det finns en bra samling av olika CSS-nollställningsmetoder. Till följande demonstreras några exempel på hur olika nollställningar kan se ut.

5.2.2 Kritik

Användning av nollställningar har också kritiserats. Jens O. Meiert anser följande om CSS nollställningar i sin nätartikel: *Why “Reset” Style Sheets Are Bad.* Meiert anser att användning av en färdig nollställning gör att man ofta bara skriver definieringar på nytt istället för att ändra värdet i själva nollställningen. Meiert anser också att CSS nollställningar ökar onödigt mycket på nätsidans laddningstid och att t.ex. egenskaper som outline som i eric meyers nollställning nollställs kan vara viktigt för tangentbords navigering.
(”*http://meiert.com/en/blog/20080419/reset-style-sheets-are-bad/*”, Hämtat 23.5.2010)

5.2.3 Minimalistisk nollställning

Det här är antagligen den minsta versionen man kan kalla för CSS nollställning. Den använder sig av en Asterix, på engelska wildcard väljare som betyder att värden som definieras innanför klamrarna berör alla element som finns i HTML dokumentet och nollställer deras marginaler och spaltfyllnad. Så här ser den ut:

```
* {
      padding: 0;
      margin: 0;
      }
```

5.2.4 Förtätad nollställning

Följande exempel jag valt anser Jeff starr, personen som skapat samlingen ”*A Killer Collection of Global CSS Reset Styles”* vara hans favorit. Den nollställer betydligt flera värden än det första exemplet men förblir ändå relativt kort.

```
* {
      vertical-align: baseline;
      font-weight: inherit;
      font-family: inherit;
      font-style: inherit;
      font-size: 100%;
      border: 0 none;
      outline: 0;
      padding: 0;
      margin: 0;
      }
```

5.2.5 Eric Meyers nollställning

Ett av de mest kända nollställningar är Eric Meyers nollställning. Eric Meyer har skrivit flera böcker om CSS och är känd för bland annat sina föreläsningar om CSS och webb-standarder. Erics version används mycket i färdiga underlag, bl.a. i Blueprint (blue-printcss.org), (Zeldman 2003: XV)

Eric Meyer anser följande om sin CSS nollställning:

> ”I don't particularly recommend that you just use this in its unaltered state in your own projects. It should be tweaked, edited, extended, and otherwise tuned to match your specific reset baseline. Fill in your preferred colors for the page, links, and so on.” (http://meyerweb.com/eric/tools/css/reset/, Hämtat 23.5.2010

Eric Meyers CSS nollställning

```
/* v1.0 | 20080212 */

html, body, div, span, applet, object, iframe,
h1, h2, h3, h4, h5, h6, p, blockquote, pre,
a, abbr, acronym, address, big, cite, code,
del, dfn, em, font, img, ins, kbd, q, s, samp,
small, strike, strong, sub, sup, tt, var,
b, u, i, center,
dl, dt, dd, ol, ul, li,
fieldset, form, label, legend,
table, caption, tbody, tfoot, thead, tr, th, td {
	margin: 0;
	padding: 0;
	border: 0;
	outline: 0;
	font-size: 100%;
	vertical-align: baseline;
	background: transparent;
}
body {
	line-height: 1;
}
ol, ul {
	list-style: none;
}
blockquote, q {
	quotes: none;
}
blockquote:before, blockquote:after,
q:before, q:after {
	content: '';
	content: none;
}

/* remember to define focus styles! */
:focus {
	outline: 0;
}

/* remember to highlight inserts somehow! */
ins {
	text-decoration: none;
}
del {
	text-decoration: line-through;
}

/* tables still need 'cellspacing="0"' in the markup */
table {
	border-collapse: collapse;
	border-spacing: 0;
}
```

5.3 Validering

Genom att validera märkspråket och stilmallen man skrivit eliminerar man eventuella skrivfel och olagligheter man gjort som bryter emot dokument deklarationen man valt att använda. Även om valideringen inte i sig själv är någon garanti för att undvika tolkningsskillnader mellan webbläsare, gör det i varje fall arbetet lättare då man skall uppdatera nätsidan eller hitta fel som förorsakar skillnader mellan webbläsare.

Valideringen för märkspråket kan utföras vid webbadressen ”*http://validator.w3.org*” och valideringen för stilmallen kan utföras vid webbadressen ”*http://jigsaw.w3.org/css-validator/*”.

Om man använder sig av kompromisser för att stöda äldre webbläsare kan det vara omöjligt att validera nätsidan, det att en nätsida inte valideras behöver inte betyda att sidan inte skulle fungera mellan olika webbläsare.

Som exempel kan man ta t.ex. Hufvudstadsbladets internetsida ”*http://www.hbl.fi/”* som 16.5.2010 enligt ”*http://validator.w3.org*” hade 613 fel i sitt märkspråk av dokumenttypen XHTML 1.0 Strict. Då man ser på sidans källkod kan man genast notera att t.ex. vissa av <meta> elementen är skrivna med små bokstäver och vissa med stora. Redan detta bryter mot reglerna om hur XHTML får skall skrivas eftersom man i XHTML får använda endast små bokstäver i märkspråket. Poängen är dock den att nätsidan ändå fungerar som den skall i de flesta webbläsare.

5.4 Testa nätsidorna

Om man är osäker om hur olika webbläsare presenterar nätsidor under olika operativsystem kan man använda sig av t.ex. ”*http://browsershots.org*” eller ”*http://www.browsrcamp.com*”.

5.5 Korrigera fel

Tyvärr måste man ännu år 2010 beakta en sådan webbläsare som IE6 om man designar nätsidor. Det att ungefär mellan 10 till 20 procent av de som surfar på nätet antagligen fortfarande använder sig av IE6 innebär att man som webbdesigner även bör beakta dessa användare då man skapar nätsidor.

Genom att använda sig av konditionala kommentarer för Internet Explorer kan man definiera en presentationsstil som bara berör t.ex. Internet Explorer 6.

HTML:

```
<!--[if IE6]>
        <link href="ie.css" rel="presentationsstil.css" type="text/css" />
<![endif]-->
```

(”*http://www.catswhocode.com/blog/15-techniques-and-tools-for-cross-browser-css-coding*” Hämtat: 17.5.2010)

6 SLUTSATS

Det är synd att så många fortfarande använder sig av den gamla webbläsaren IE6. Det att man ännu måste skriva webbläsarspecifika definitioner bromsar ned på utvecklingen och skapar en massa onödigt jobb för webbdesigners. IE6 kom ut för 9 år sedan, det är en väldigt lång tid om man tänker efter hur mycket internet har utvecklats sedan detta.

Om man ser emot framtiden och ser på de moderna webbläsare som stöder CSS 2.1 kan man skapa nätsidor som fungerar bra mellan de olika webbläsarna utan att behöva använda sig av webbläsarspecifika definitioner.

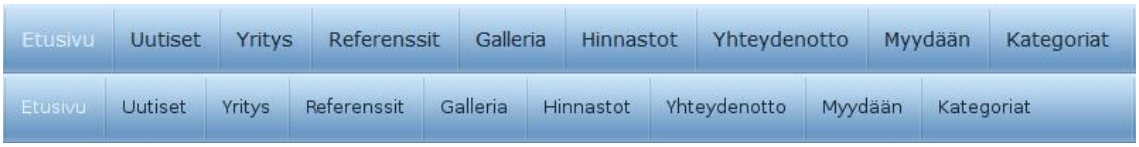
Genom att känna till vad som förorsakar tolkningsskillnader mellan webbläsare, undviker man lättare själv att göra dem. Då man använder sig av webbstandarder som olika webbläsare stöder och noggrant specificerar hur man vill att webbläsarna skall presentera en nätsida, finns det väldigt lite rum för webbläsare att göra egna tolkningar.

En viktig sak jag insett under min forskning är att det är en viss skillnad med att skapa en cross-browser kompatibel nätsida och det jag egentligen var ute efter i mitt arbete. Jag har insett att en cross-browser kompatibel nätsida innebär att nätsidan fungerar i olika webbläsare, men att det absolut inte betyder att den skulle behöva presenteras pixelnoggrant i olika webbläsare, utan endast att den stöder de olika webbläsare för att få fram budskapet. En cross-browser kompatibel nätsida kan t.ex. ha en skild presentationsstil för webbläsare på datorer och mobila webbläsare.

Jag hade tidigare en tanke om att man skulle kunna skapa en nätsida som är pixelnoggrann i olika webbläsare men insåg att det var så gott som omöjligt och hjälper inte en webbdesigner att spara tid, snarare tvärtom. I stället ansåg jag att man kunde tänka sig så att vissa element är dynamiska och vissa statiska.

Jag tänker mig så att allting som i en nätsida är statiskt i praktiken är sådant som presenteras pixelnoggrant mellan webbläsare. Det kan vara frångan om t.ex. bilder eller grafik som skapats med CSS. Eftersom man inte fullständigt kan kontrollera hur fonter och formulär beter sig, behandlar jag dem som dynamiska. Genom att i designskedet ta i beaktande hur fonter och formulär beter sig kan man undvika att dessa bryter sönder nätsidan genom att t.ex. placera dem i flexibla containers.

I följande exempel ville en kund att navigeringslänkarna på kundens nätsida skulle placeras så att de skulle fylla navigeringsbalken jämt. Då sidan visades med en webbläsare som var installerad under ett Linux operativsystem bröts sidans design eftersom operativsystemet använde sig av en font som var mycket kortare än den som ursprungligen var planerad för designen.



Figur 13. Övre balken är en skärmbild från Windows7 med Firefox som webbläsare och den undre balken är en skärmbild från Linux med webbläsaren Firefox.

Det man kan konstatera ur detta är att man helst skall undvika att använda fonter som en del av designen.

6.1 Frågeställningarna

1. Kan man med dagens webbtekniker skapa nätsidor som är pixelnoggranna mellan moderna webbläsare?

På basen av den här forskningen kan jag konstatera att detta inte är möjligt att uppnå med endast HTML och CSS om man inte lämnar bort de dynamiska elementen och endast använder sig av grafik. Med JavaScript och mycket arbete kan man troligtvist skapa en nätsida som är så gott som pixelnoggrann mellan olika webbläsare.

2. Vilka faktorer inverkar på hur olika webbläsare tolkar nätsidor?

De mest centrala faktorer jag kommit till i min forskning är att en webbläsare tolkar en nätsida enligt de instruktioner man ger den, då en webbläsare själv måste välja hur den skall presentera någonting i en nätsida finns det en risk för tolkningsskillnader. Äldre webbläsare förstår sig inte lika bra på webbstandarder som nya webbläsare. Fonter och formulär är väldigt bundna till den webbläsare som används och kan skilja sig en aning mellan olika webbläsare.

3. Hur kan man minska på tolkningsskillnader mellan webbläsare?

Med att använda sig av webbstandarder, validering och CSS-nollställningar kan man betydligt minska på de vanligaste orsaker som förorsakar skillnader mellan hur webbläsare tolkar nätsidor. Man bör även ta i beaktande hur fonter och formulär varierar mellan olika webbläsare för att undvika eventuella överskridningar.

6.2 Slutord

Med den här forskningen har jag kommit till den slutsatsen att jag anser att det inte lönar sig att skapa nätsidor som är pixelnoggrann mellan olika webbläsare, då man använder sig av noggranna HTML och CSS. Det hur fonter och formulär beter sig mellan olika webbläsare kan man inte kontrollera till fullo. Om man absolut vill skapa nätsidor som är pixelnoggranna mellan olika webbläsare måste man använda sig av lösningar som befinner sig utanför HTML och CSS, detta kräver igen mera tid.

Genom att använda sig av XHTML med strikt DTD och CSS-nollställning har man en bra grund att börja bygga upp en nätsida med bra stöd ännu långt i framtiden. Genom att behandla element som statiska och dynamiska kan man lättare förutse hur de beter sig i olika webbläsare och kan designa nätsidan från början på ett sådant sätt att det inte uppstår konflikter.

Då jag började den här forskningen viste jag inte riktigt vad jag försökte uppnå. Jag ville hitta ett svar på hur man kan skapa cross-browser kompatibla nätsidor utan att behöva lägga ned en massa tid på att korrigera webbläsarspecifika konflikter i efterhand. Det viktigaste med den här forskningen har dock varit att jag lärt mig att tänka på ett helt nytt sätt då jag designar nätsidor.

KÄLLOR

Litteratur:

Sawyer McFarland, David. 2009. CSS: The missing manual 2 uppl.
USA: O'Reilly Media, Inc. 538 s. ISBN 978-0-596-80244-8.

Murphy Christopher, Persson Nicklas. 2009. HTML and CSS Web Standards Solutions:
A Web Standardistas' Approach. USA 397 s. ISBN 978-1-4302-1607-0.

Zeldman.Jeffrey, 2003. Designing with Web Standards.
USA. 415 s. ISBN 0-7357-1201-8.

Internetkällor:

w3schools.com - Why use HTML 4.0?
(”*http://www.w3schools.com/html/html_whyusehtml4.asp*”, Hämtat: 17.5.2010)

w3schools - CSS Introduction
(”*http://www.w3schools.com/css/css_intro.asp* ”, Hämtat 23.5.2010)

w3schools.com - Browser Statistics
(”*http://www.w3schools.com/browsers/browsers_stats.asp*”, Hämtat 23.5.2010)

w3schools.com - XML Tutorial
(“w3schools.com/xml/”, Hämtat: 8.5.2010)

w3schools.com - XHTML Tutorial
(“www.w3schools.com/xhtml/, Hämtat: 7.5.2010)

w3schools.com - HTML <!DOCTYPE> Declaration
(”*http://www.w3schools.com/tags/tag_doctype.asp*”, Hämtat: 10.5.2010)

w3schools.com - CSS Id and Class
(”*http://www.w3schools.com/css/css_id_class.asp*”, Hämtat:17.5.2010)

w3schools.com - CSS Grouping and Nesting Selectors
(”http://www.w3schools.com/css/css_grouping_nesting.asp”, Hämtat: 23.5.2010)

netmarketshare.com
(”*http://www.netmarketshare.com/browser-market-share.aspx?qprid=1*”, Hämtat 23.5.2010)

sixrevisions.com - CSS Tip #1: Resetting Your Styles with CSS Reset
(”*http://sixrevisions.com/css/css-tips/css-tip-1-resetting-your-styles-with-css-reset/*”, Hämtat 14.5.2010)

onyx-design.net
(”http://www.onyx-design.net/other/cross-browser-compatible-in-5-steps/”, hämtat 17.4.2010)

perishablepress.com - A Killer Collection of Global CSS Reset Styles
(”*http://perishablepress.com/press/2007/10/23/a-killer-collection-of-global-css-reset-styles/*, Hämtat 23.5.2010)

sv.wikipedia.org - Cascading Style Sheets
(”*http://sv.wikipedia.org/wiki/Cascading_Style_Sheets*”, Hämtat: 17.5.2010)
sv.wikipedia.org - HTML
(”*http://sv.wikipedia.org/wiki/HTML*”, Hämtat 17.4.2010)

sv.wikipedia.org - märkspråk
(“http://sv.wikipedia.org/wiki/M%C3%A4rkspr%C3%A5k”, Hämtat 17.4.2010)

meyerweb.com – CSS reset
(”*http://meyerweb.com/eric/tools/css/reset/*”, Hämtat 17.5.2010)

meyerweb.com – formal weirdness
(“*http://meyerweb.com/eric/thoughts/2007/05/15/formal-weirdness/*”, Hämtat 1.6.2010)

blueprint.org
(”*http://www.blueprintcss.org/*”, Hämtat 17.4.2010)

meiert.com
(“*http://meiert.com/en/blog/20080419/reset-style-sheets-are-bad*/”, Hämtat 17.4.2010)

joedolson.com
(”*http://www.joedolson.com/articles/2008/03/what-is-cross-browser-compatibility/”,* Hämtat 24.5.2010)

catswhocode.com
(”*http://www.catswhocode.com/blog/15-techniques-and-tools-for-cross-browser-css-coding*” Hämtat: 17.5.2010)

netmechanic.com - Browser Compatibility Tutorial
(”*http://www.netmechanic.com/products/Browser-Tutorial.shtml*”, Tom Dahm. Hämtat 1.6.2010)

quirksmode.org
(”*http://www.quirksmode.org/compatibility.html*”, Hämtat 23.5.2010)

BILAGOR

Gemensam stil för fonter.

HTML dokumentet som användes för att få länkarna att se exakt lika ut i tre olika webb-läsare.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Untitled Document</title>
<style type="text/css">
<!--
* {
	margin: 0;
	padding: 0;
	border: 0;
	outline: 0;
	font-size: 100%;
	vertical-align: baseline;
	background: transparent;
}

a {
	text-decoration: underline;
	font-family: Arial, Helvetica, sans-serif;
	font-size: 14px;
	color: #00C;
	line-height: 14px;

}
div {
	line-height: 14px;
	font-size: 14px;
}

-->
</style>
</head>

<body>

<div><a href="">Länk</a></div>

</body>
</html>
```

Formulärtestet

HTML dokumentet som användes för att testa hur olika webbläsare klarar av att presentera select elementet.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Untitled Document</title>
<style type="text/css">
<!--
body form select {
	border: 2px solid #000;
}
-->
</style>
</head>

<body>
<form>
<select name="">
<option>TEST 1</option>
<option>TEST 2</option>
<option>TEST 3</option>
<option>TEST 4</option>
</select>
</form>
</body>
</html>
```

Font skillnader.

HTML-dokumentet som användes för att gemföra fonter mellan olika webbläsare och operativsystem.

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<title>Verdana TEST</title>

<style type="text/css">
<!--

/* v1.0 | 20080212 */

html, body, div, span, applet, object, iframe,
h1, h2, h3, h4, h5, h6, p, blockquote, pre,
a, abbr, acronym, address, big, cite, code,
del, dfn, em, font, img, ins, kbd, q, s, samp,
small, strike, strong, sub, sup, tt, var,
b, u, i, center,
dl, dt, dd, ol, ul, li,
fieldset, form, label, legend,
table, caption, tbody, tfoot, thead, tr, th, td {
	margin: 0;
	padding: 0;
	border: 0;
	outline: 0;
	font-size: 100%;
	vertical-align: baseline;
	background: transparent;
}
body {
	line-height: 1;
}
ol, ul {
	list-style: none;
}
blockquote, q {
	quotes: none;
}
blockquote:before, blockquote:after,
q:before, q:after {
	content: '';
	content: none;
}

/* remember to define focus styles! */
:focus {
	outline: 0;
}

/* remember to highlight inserts somehow! */
ins {
	text-decoration: none;
}
del {
	text-decoration: line-through;
```

```
}

/* tables still need 'cellspacing="0"' in the markup */
table {
	border-collapse: collapse;
	border-spacing: 0;
}

#a {font-family: Verdana;}
#b {font-family: Georgia;}
#c {font-family: "Courier New";}
#d {font-family: Arial;}
#e {font-family: Tahoma;}
#f {font-family: "Trebuchet MS";}
#g {font-family: "Arial Black";}
#h {font-family: "Times New Roman";}
#i {font-family: "Palatino Linotype";}
#j {font-family: "Lucida Sans Unicode";}
#k {font-family: "MS Serif";}
#l {font-family: "Lucida Console";}
#m {font-family: "Comic Sans MS";}

#a,#b,#c,#d,#e,#f,#g,#h,#i,#j,#k,#l,#m {
	font-size: 14px;
	font-style: normal;
	font-weight: normal;
	font-variant: normal;
	text-transform: none;
	color: #333;
	line-height: 14px;
	border-top-width: 1px;
	border-top-style: solid;
	border-right-style: solid;
	border-bottom-style: solid;
	border-left-style: solid;
	border-top-color: #CCC;
	border-right-color: #CCC;
	border-bottom-color: #999;
	border-left-color: #CCC;
}
-->
</style>
</head>
<body>

<div id="a">Verdana</div>
<div id="b">Georgia</div>
<div id="c">Courier New</div>
<div id="d">Arial</div>
<div id="e">Tahoma</div>
<div id="f">Trebuchet MS</div>
<div id="g">Arial Black</div>
<div id="h">Times New Roman</div>
<div id="i">Palatino Linotype</div>
<div id="j">Lucida Sans Unicode</div>
<div id="k">MS Serif</div>
<div id="l">Lucida Console</div>
<div id="m">Comic Sans MS</div>

</body>
</html>
```