Difficulties maintaining separation of structure and presentation while using a browser based WYSIWYG-editor

Simon Rönnqvist

Arcada

Media culture

Helsinki 2007

EXAMINATION OF THESIS WORK

Examination language: English

Field of Study
The thesis is concerned with problematic aspects of the relationship between structure and content in the design of web pages.

Structure and Content
The thesis begins by looking at the issues to be discussing and explaining, from a theoretical perspective, why they are important. It looks at the history of presentation on the world wide web, and at the tools that have been used to render this presentation.

The second section of the thesis is a detailed case study that shows the ways in which presentational theory manifests itself in day-to-day practice. It demonstrates the ambiguity and confusion that can result from the present tool sets.

The third section is an analysis of the case study. It examines the ambiguities that appear in the mark-up of the site being studied; traces their origins; and draws conclusions about the intentions of the author of the content, and the consequences of the choices made.

Finally the thesis offers some pointers for further study, and a set of ideas for ways forward.

Analysis of the thesis
The issue that has been chosen for study has been of historical importance, and continues to cause problems for web designers. Arguably with the growth of tools allowing for more user input to sites the problem is even more important, and more urgent, now than it has been previously.

This thesis has a clear aim, and succeeds admirably in pursuing this to a successful conclusion. It narrows down a broad range of questions and focuses crisply on a single but telling example; and yet is still able to draw general conclusions from this.

The conclusions that are drawn, and the way that they are presented, means that this thesis should have a useful life within Arcada, and elsewhere, for some time after the student has graduated.

Assessment of the thesis
The work that has been done in this thesis has involved establishing a theoretical framework; drawing up a research question; establishing research to answer that question; and documenting the results. This has all been achieved to a very high standard, and we are therefore happy to award this thesis a grade of **Excellent**.

Helsinki 01.09.2007

[illegible]

John Grönvall
Handledare (Supervisor)

[illegible]

Owen Kelly
Granskare (Examiner)

DEGREE THESIS	
Arcada	
Degree Programme:	Media Culture
Identification number:	2170
Author:	Simon Rönnqvist
Title:	Difficulties maintaining separation of structure and presentation while using a browser based WYSIWYG-editor
Supervisor:	John Grönvall
Commissioned by:	

Abstract:

According to recommendations by the World Wide Web Consortium (W3C) one should structure web content (using XHTML) according to semantic meaning and separately control how certain kinds of elements should be presented (using CSS). WYSIWYG-editors in CMS-environments tend to cause inconsistencies because they allow their users to within the content define the looks of each element that they are editing. A WYSIWYG which instead is intended for structuring content semantically is the Bitflux Editor. The thesis showed through a case study how a user of the Bitflux Editor was mislead by its WYSIWYG-nature to mark up content according to presentation instead of semantic meaning. (E.g. headings were used to achieve certain text styles.) Thus it was concluded that the WYSIWYG-concept is unsuitable for CMS-usage and for web content in general, since it causes inconsistency. The case study examined the troubadour Håkan Streng's web site, which represented a rather typical web site with mostly text content. Also a usability problem in WYSIWYG-editors intended for structural content was identified; they show presentation but expect their users to think according to structure. It was concluded that the solution to both this usability problem and to the problem with inconsistently marked up content could be having purely structural editors, that instead show the document structure while editing. One such concept is called WYSIWYM (What You See Is What You Mean), which could be tried as a replacement for WYSIWYG:s in CMS-contexts.

Keywords:	HTML, CSS, CMS, WYSIWYG
Number of pages:	47
Language:	English
Date of acceptance:	18.06.2007

EXAMENSARBETE	
Arcada	
Utbildningsprogram:	Mediekultur
Identifikationsnummer:	2170
Författare:	Simon Rönnqvist
Arbetets namn:	Svårigheter med att upprätthålla separation av struktur och presentation vid använding av en webbläsarbaserad WYSIWYG-editor
Handledare:	John Grönvall
Uppdragsgivare:	

Sammandrag:

Enligt World Wide Web Consortiums (W3C) rekommendationer borde webbinnehåll struktureras (med XHTML) enligt semantisk betydelse, medan man separat (med CSS) bör definiera hur olika typer av element skall presenteras. WYSIWYG-editorer i CMS-miljöer tenderar att orsaka inkonsekvent innehåll för att de tillåter sina användare definiera utseende skilt för varje element i det innehåll som de redigerar. En WYSIWYG som istället är avsedd för att strukturera innehåll enligt semantik är Bitflux Editor. Examensarbetet visar genom en fallstudie hur en användare av Bitflux Editor vilseleds av dess WYSIWYG-natur till att strukturera innehållet enligt presentation istället för semantisk betydelse. (T.ex. rubriker användes för att åstadkomma vissa textstilar.) Således drogs slutsatsen att WYSIWYG-konceptet är olämpligt för CMS-användning och webbinnehåll i allmänhet, eftersom det orsakar inkonsekvent strukturering av innehållet. I fallstudien undersöktes trubaduren Håkan Strengs webbsajt, vilken representerade en ganska typisk webbsajt bestående av mestadels textinnehåll. Även ett användbarhetsproblem i WYSIWYG-editorer avsedda för semantiskt strukturerat innehåll identifierades; de visar presentation medan användaren förväntas tänka enligt semantisk struktur. Det konstaterades att en lösning till detta användbarhetsproblem såväl som problemet med inkonsekvent strukturerat innehåll kunde vara att använda rent strukturorienterade editorer, som istället skulle visa semantisk dokumentstruktur under redigeringsprocessen. Ett sådant koncept är WYSIWYM (What You See Is What You Mean), vilket kunde testas som ersättare för WYSIWYG i CMS-miljöer.

Nyckelord:	HTML, CSS, CMS, WYSIWYG
Sidantal:	47
Språk:	engelska
Datum för godkännande:	18.06.2007

Dedication

Thanks to my wife, son, colleagues, teachers and friends at Arcada for being around as a support throughout the time that I've learned all that's presented here and more. This thesis would not have been the same without you!

Contents

1 Introduction

1.1 Basic concepts

Content management systems (CMS) are often equipped with so called WYSIWYG[1]-editors (from now on referred to as WYSIWYG). A WYSIWYG works in some ways like an ordinary word processor, striving to show the content editor what the end result would look like in real-time while editing.

A WYSIWYG often lets the content editor separately define the looks of each element. If this kind of functionality is used, it leads to poor separation of document structure[2] and presentation[3].

When defining presentation separately one defines how certain kinds of structural elements should be presented (in certain kinds of media if you wish). In practice poor separation of structure and presentation can lead to inconsistent looks of a site. It can also lead to problems when redesigning a site, since the presentational control isn't centralized. In some cases one would also want different styling for different presentational media, which is almost impossible if the presentation isn't defined separately.

The World Wide Web Consortium (W3C), lead by the inventor of the Web TIM BERNERS-LEE, develops web standards such as XHTML[4] (for structuring content) and CSS[5] (for defining presentation). Using web standards according to their recommendations is commonly referred to as standards based web design, or standards compliant web design. Keeping structure and presentation separate is a central part of standards based web design.

[1]What You See Is What You Get
[2]Structural elements are e.g. main headings, sub-headings and links.
[3]Presentation is how a document should be presented in a certain media, e.g. screen, print, screen readers or handheld devices.
[4]eXtensible Hyper-Text Markup Language
[5]Cascading Style Sheets

1.2 Case background

My production is the troubadour Håkan Streng's web site. The site is quite an ordinary kind of site, with mostly text content. The site utilizes a CMS equipped with a WYSIWYG for the content editor to use.

The production is an attempt to maintain separation of structure and presentation by using an in this respect strict WYSIWYG. One of the few web based WYSIWYG:s of today that validate the structure of the content when editing is the Bitflux Editor, and it is therefore used in the production. It is used within Flux CMS, the web based CMS for which it's primarily developed.

1.3 Goal

Due to the live validation the Bitflux Editor manages to maintain a valid XHTML 1.0 Strict[6] document structure in the production mentioned above. XHTML 1.0 Strict forbids use of obsolete[7] presentational markup[8]. The central argument of my thesis is that separation of structure and presentation cannot be maintained purely by technical means, while using a WYSIWYG such as the Bitflux Editor.

My analysis will consist of mapping out what kinds of failures to maintain a separation of structure and presentation occurred, in spite of the strict nature of the Bitflux Editor. I will later also discuss possible ways to avoid these problems. The target group for this thesis is people working with standards based web design for CMS-based sites, in other words basically anyone making a modern web site. Basic knowledge of XHTML is assumed, though some brief explanations are given on the XHTML-terminology needed to understand the thesis.

[6]XHTML 1.0 The Extensible HyperText Markup Language `http://www.w3.org/TR/xhtml1/`

[7]Some versions of XHTML's predecessor HTML was to some extent intended to specify presentational aspects of a web page. This kind of features have later been deprecated and are no longer allowed at all in XHTML 1.0 and 1.1 Strict.

[8]XHTML, the predecessor HTML and even it's predecessor SGML (used in print) are commonly referred to as markup.

2 Theory

2.1 Initial visions pointing out the future

To realize how to build sustainable web content and applications one needs to be aware of the initial visions that still point the direction in which the Web is developing. These visions are well described by the inventor of the web, TIM-BERNERS LEE:

> I have a dream for the Web...and it has two parts.
>
> In the first part, the Web becomes a much more powerful means for collaboration between people. I have always imagined the information space as something to which everyone has immediate and intuitive access, and not just to browse, but to create. The initial *WorldWide Web* program opened with an almost blank page, ready for the jottings of the user. Robert Cailliau and I had a great time with it, not because we were looking at a lot of stuff, but because we were writing and sharing out ideas. Furthermore, the dream of people-to-people communication through shared knowledge must be possible for groups of all sizes, interacting electronically with as much ease as they do now in person.
>
> In the second part of the dream, collaborations extend to computers. Machines become capable of analyzing all the data on the Web – the content, link, and transactions between people and computers. A "Semantic Web", which should make this possible, has yet to emerge, but when it does, the day-to-day mechanisms of trade, bureaucracy, and our daily lives will be handled by machines talking to machines, leaving humans to provide the inspiration and intuition. The intelligent "agents" people have touted for ages will finally materialize. This machine-understandable Web will come about through the implementations of a series of technical advances and social agreements that are now beginning.

> Once the two-part dream is reached, the Web will be a place where the whim of a human being and the reasoning of a machine coexist in an ideal, powerful mixture.
>
> [Berners-Lee(2000), p. 157-158]

Today we have the concept of "Web 2.0"[1], which is a collection of many different concepts that characterize modern web applications. Among these things are features for collaboration between people and machine readable content, letting content flow automatically in between applications. While this is a step in the right direction, it's not all that BERNERS-LEE had in mind. The Semantic Web is still only in its infancy since important technologies especially made for structuring machine readable content, such as RDF (Resource Definition Format), are not yet used on the vast majority of the web sites of today.

Some kinds of CMS-implementations could be compared to the tools BERNERS-LEE describe, letting an ordinary web user edit content. As mentioned in 1.1 on page 9 many CMS:es are equipped with WYSIWYG:s that quite often let users produce poorly structured web content. In a web evolving towards being semantic, ordinary human readable web content should be structured in a semantic way as well, and not a presentational. (Not only RDF but also XHTML is to be used semantically.) Semantically structured content means that the markup tells what the different parts of the content are, and not how they should be presented. For instance it can tell that a certain part of content is a main heading, but not how it should look. This will be described further in 2.3 on page 14, but before moving there we'll take a look at this from a historical perspective.

2.2 History

One of the foremost CSS experts ERIC MEYER describes how web design came to be, in the days before the proper tool for the job (CSS) was available:

> Back in the dimly remembered, early years of the Web (1990-1993), HTML was a fairly lean language. It was composed almost entirely of structural elements that were useful for describing things like paragraphs, hyperlinks, lists, and headings. It had nothing even remotely approaching tables, frames, or the complex markup we assume is necessary to create web pages. HTML was originally intended to be a structural markup language, used to describe the various parts of a document; very little was said about how those parts

[1]O'Reilly article defining Web 2.0 `http://www.oreillynet.com/pub/a/oreilly/tim/news/2005/09/30/what-is-web-20.html`

> should be displayed. The language wasn't concerned with appearance — it was just a clean little markup scheme.
>
> [Meyer(2006), p. 1]

Instead the early browsers controlled the page styling internally only, so that the styling for each kind of element was up to the user to decide, not the content author who only was equipped with tools to structure the content. [Lie and Bos(1999)]

> As the number of sites increased, so did the demand for new HTML elements that would each perform a specific function. Authors started demanding that they be able to make text boldfaced or italicized.
>
> At the time, HTML wasn't equipped to handle those sorts of desires. You could declare a bit of text to be emphasized, but that wasn't necessarily the same as being italicized — it could be boldfaced instead, or even normal text with a different color, depending on the user's browser and preferences. There was nothing to ensure that what the author created was what the reader would see.
>
> As a result of these pressures, markup elements like <FONT> and <BIG> started to creep into the language. Suddenly, a structural language started to become presentational.
>
> [Meyer(2006), p. 2]

Not only did browsers start implementing some purely presentational markup. In 1995, the typographer and early web designer, DAVID SIEGEL published an on-line tutorial explaining techniques to work around the limitations of HTML, enabling design of actual magazine-like layouts on the Web. The year after he published his book "Creating Killer Websites", which was the first book that seriously treated the subject web design. Unfortunately this was the advent of many common practices that seriously conflict the initial ideas of HTML as a structural language. [Zeldman(2001), p. 117-118]

Even today we can see tables and invisible spacer images being used to build up layouts, although using CSS is a much more convenient way to make layouts, without using presentational markup.

Since 1996 the usage of CSS has been a recommendation by the W3C. [Meyer(2006), p. 3] And since year 2000 all new mainstream browsers have had sufficient support for web standards such as CSS, to allow web designers to build up layouts using CSS instead of presentational markup. [Zeldman(2001), p. 121-122]

Browsers with insufficient support for CSS-layouts are now very rarely used, since their use died out over the last couple of years. Even for those older browsers the content of CSS-designed pages can still be left accessible (though unstyled). [Zeldman(2006), p. 171-172] Therefore there is seldom any valid reason besides lack of up-to-date knowledge not to embrace CSS-design and semantically structured markup.

2.3 Separation of structure and presentation

2.3.1 Why?

The core reasons to keep structured content and presentation separate are to keep content accessible to a wide variety of agents (usually browsers) and to keep content easily reusable "across multiple delivery contexts", as stated in W3C:s recommendations:

> The Web is a heterogeneous environment where a wide variety of agents provide access to content to users with a wide variety of capabilities. It is good practice for authors to create content that can reach the widest possible audience, including users with graphical desktop computers, hand-held devices and mobile phones, users with disabilities who may require speech synthesizers, and devices not yet imagined. Furthermore, authors cannot predict in some cases how an agent will display or process their content. Experience shows that the separation of content, presentation, and interaction promotes the reuse and device-independence of content; this follows from the principle of orthogonal specifications (§5.1) `http://www.w3.org/TR/2004/REC-webarch-20041215/#orthogonal-specs`.
>
> This separation also facilitates reuse of authored source content across multiple delivery contexts. Sometimes, functional user experiences suited to any delivery context can be generated by using an adaptation process applied to a representation that does not depend on the access mechanism. For more information about principles of device-independence, see [DIPRINCIPLES] `http://www.w3.org/TR/2004/REC-webarch-20041215/#DIPRINCIPLES`.
>
> [Group(2004)]

In other words pages need to have well structured content in "...today's more sophisticated web, where pages are frequently assembled by publishing tools and content must flow back and forth from database to web page to wireless device to print...", as described by web standards expert and co-founder of The Web Standards Project[2] JEFFERY ZELDMAN. [Zeldman(2006), p. 105]

[2]The Web Standards Project or WaSP is a grassroots coalition of web designers promoting web standards, one of its greatest achievements was helping to end the Browser Wars by persuading Microsoft and Netscape to support the same technologies in their browsers. `http://www.webstandards.org`

Today also many countries worldwide either require or recommend that web pages of governmental institutions and such should comply with certain accessibility guidelines, usually WCAG 1.0[3] by the W3C. [Initiative(2006)] Many big companies such as banks also seem to use standards based web design, to not shut out any potential customers and to most often even ensure them a smooth web experience.

Complying with accessibility guidelines such as WCAG 1.0 includes separation of structure and presentation. [WCA(1999)] Working drafts of the upcoming version WCAG 2.0 have been published on the Web Accessibility Initiative web site `http://www.w3.org/WAI/`.

2.3.2 Concepts

Slightly different words are used for the same concepts. Some talk of separation of *content* and *presentation*, some of *structure* and *presentation*. To be precise one could talk about 'structured content', but to stick with an established terminology we'll go with just *structure*, as used by ZELDMAN. He also describes *behavior* as one third component, which basically means scripted client side features, but since the web pages discussed in this thesis lack client side scripting it is not discussed any further. (In some cases *separation of content and presentation* is also used for server-side separation, when storing data separately in databases. However this thesis is only concerned with the pages fed to the client.)

[3]Web Content Accessibility Guidelines 1.0 `http://www.w3.org/TR/WAI-WEBCONTENT/`

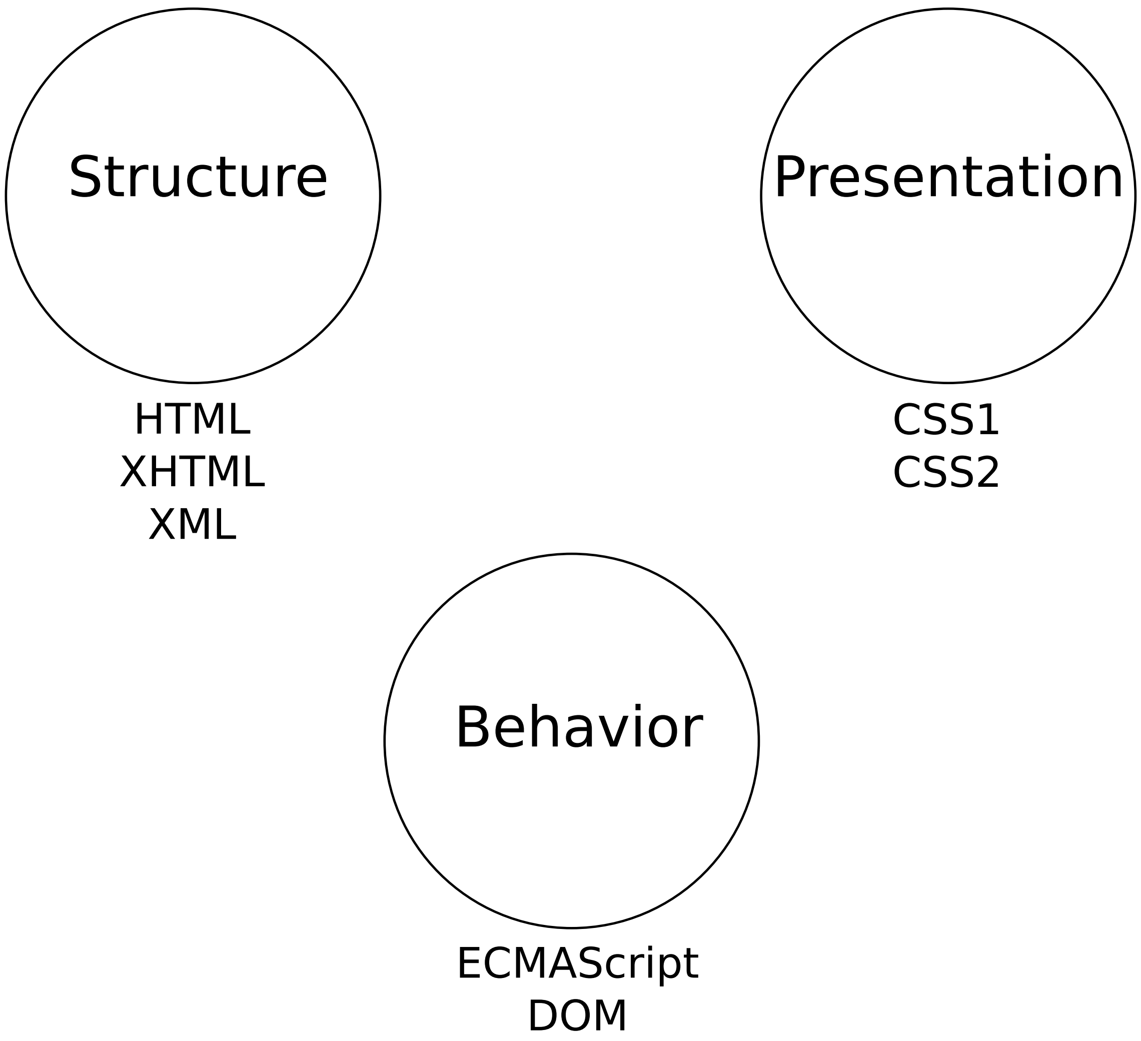


Figure 2.1: Structure presentation and behavior are the three components of web standards based pages. The technologies that build up each of the components are listed below the bubbles. [Zeldman(2006)]

2.3.3 Semantic markup

Since this thesis concentrates on the content generated by a web based WYSIWYG inside of a CMS, and not separate presentation in the form of CSS (which is made by the web designer), CSS is only presented as a concept while as XHTML is presented a bit further here.

As stated repeatedly earlier separation of structure and presentation is important. Understanding the concept presentation is pretty straight forward, but what is meant by structuring content semantically?

> If you treat HTML or XHTML as a document-formatting tool, you will be sorely disappointed. There is simply not enough capability built into the languages to allow you to create the kinds of documents you might whip up with

> tools such as FrameMaker and Word. Attempts to subvert the supplied structuring elements to achieve specific formatting tricks seldom work across all browsers. In short, don't waste your time trying to force HTML and XHTML to do things they were never designed to do.
>
> Instead, use HTML and XHTML in the manner for which they were designed: indicating the structure of a documents so that the browser can then render its content appropriately. HTML and XHTML are rife with tags that let you indicate the semantics of your document content, something that is missing from or often badly implemented in word processors and page-layout programs. Create your documents using these tags and you'll be happier, your documents will look and work better, and your readers will benefit immensely.
>
> [Musciano and Kennedy(2006), p. 10]

Web browsers have their default ways of displaying different kinds of content, however one cannot rely on this to be the same in each and every browser. Therefore controlling the looks of a certain element is much more reliable using CSS. Besides that, defining elements according to their default looks in certain browsers breaks their semantic value. [Zeldman(2006), p. 85, 173]

Presentationally used markup[4]

As mentioned in 2.1 semantically structured content means that the markup tells what the different parts of the content are, and not how they should be presented. Semantic markup could be thought of as a kind of meta-data (data about data).

> We are all used to thinking that <h1> means big, <li> means bullet, and <blockquote> means, "indent this text". Most of us have scribbled our share of HTML that uses structural elements to force presentational effects.
>
> Along the same lines, if a designer wants all headlines to be the same size, she might set all her headlines as <h1>, even though doing so makes no structural sense and is the kind of thing usability consultant Jakob Nielsen would call a sin if he weren't too busy worrying about link colors...
>
> [Zeldman(2006), p. 172]

Using *<br />* to add space above or under other elements is also not a semantic way of using markup, this should be achieved using CSS instead. [Zeldman(2006), p. 172]

[4]This section assumes basic knowledge of HTML or XHTML, please refer to 2.3.4 on the following page for a brief introduction

Elements chosen according to their supposed presentational properties can be considered presentational markup, even though they unlike entirely presentational and deprecated markup such as <*font*> can be used semantically. Using markup with the intention to control presentation is in fact breaking the separation of structure and presentation.

2.3.4 XHTML used in the case study

Although basic knowledge of XHTML is assumed one can indeed follow the case study by being familiar with the concepts previously presented, along with just the few XHTML elements. Therefore those few XHTML elements are presented here, along with a very brief explanation of how XHTML (and in fact other XML) documents are structured:

All of the XHTML elements below except for *line break* and *image* wrap content. Wrapping means that they consist of one opening tag <*something*> and a closing tag <*/something*>, with the wrapped content in between the tags. The wrapped content can also contain other tags. XHTML-tags (also referred to as markup) can also contain attributes according to this format <*sometag someattribute="something"*>, as far as this thesis is concerned the attributes in the code samples can be ignored, since they're not discussed any further.

<p>	paragraph - wraps paragraphs
<br />	line break - used to enforce line breaks in certain places in a text
<h*n*>	heading - wraps headings, *n* is a number representing the heading level, e.g. <*h1*> for main headings, <*h2*> for sub-headings and so forth
<em>	emphasize - wraps emphasized words or phrases
<strong>	strong emphasize - wraps strongly emphasized words or phrases
<ul>&<li>	unordered lists - <*ul*> wraps the whole unordered list (normally presented as a bulleted list) and <*li*> wraps each list item
<a>	anchor - wraps linked words
<img />	image - refers to an image document that will be shown where the tag is placed

2.4 WYSIWYG inherited from print

2.4.1 History

The WYSIWYG-concept was initially invented for the Pre-Press industry, long before the Web was around. Here are some historical notes as described in Wikipedia:

- Before the invention of WYSIWYG, all text and control characters appeared in the same typeface and style with little indication of layout (margins, spacing, etc.). Users were required to enter code tags to indicate that some text should be in boldface, italics, or a different typeface or size. These applications used an arbitrary markup language to define the tags. Because of its simplicity, this method remains popular for some basic text editing applications.

- The phrase was originated by a newsletter published by Arlene and Jose Ramos, called WYSIWYG. It was created for the emerging Pre-Press industry going electronic in the late 1970s. After 3 years of publishing, the newsletter was sold to employees at the Stanford Research Institute in California. The first conference on the topic was organized by Jonathan Seybold and the first technology popularized at Xerox PARC during the late 1970s when the first WYSIWYG editor, Bravo, was created on the Alto. The Alto monitor (72 pixels per inch) was designed so that one full page of text could be seen and then printed on the first laser printers. When the text was laid out on the screen 72 PPI font metric files were used, but when printed 300 PPI files were used - thus one would occasionally find characters and words slightly off, a problem that continues to this day. (72 PPI came from a new measure of 72 "PostScript points" per inch. Prior to this, the standard measure of 72.27 points per inch was used in typeface design, graphic design, typesetting and printing.)

- Seybold and the researchers at PARC were simply re-appropriating a popular catch phrase of the time originated by "Geraldine", Flip Wilson's drag persona from Rowan & Martin's Laugh-In in the late 60s and then on The Flip Wilson Show, (1970-1974).

- The Apple Macintosh system was originally designed so that the screen resolution and the resolution of the dot-matrix printers sold by Apple were easily scaled: 72 PPI for the screen and 144 DPI for the printers. Thus, the on-screen output of programs such as MacWrite and MacPaint were easily translated to the printer output and allowed WYSIWYG editing. With the introduction of laser printers, resolutions deviated from even multiples of the screen resolution, making WYSIWYG harder to achieve.

- Charles Simonyi, the PARC researcher responsible for Bravo, joined Microsoft in 1981 to start development of application programs at Microsoft. Hence, Bravo can be seen as the direct ancestor of Microsoft Word.

[wik(2007)]

2.4.2 Criticism of WYSIWYG

Already in 1996 *the Seybold Report on Publishing Systems* published an article by typographer CONRAD TAYLOR that examined what WYSIWYG or more precisely DTP had done to the print industry, and what it was doing to the Web.

According to the the article damage was generally done to typography concerning tasks that were better off handled automatically rather than manually through a WYSIWYG. For example the TEX[5] typesetting system does automatic H&J (hyphenation and justification) much better than WYSIWYG-editors do, since its algorithms take the text as a whole into account when applying H&J after the text is already written. Also many other problems such as lack of automatic pagination were brought up in the article.

Better H&J was also asked for in Web browsers, a need which has not even been addressed by the major browser vendors of today. Another problem mentioned that also but not solely concerns the Web is the need for structural markup in order for the content to be easily re-publishable in different contexts. SGML (Standard Generalized Markup Language), from which HTML was later derived, was already before the breakthrough of WYSIWYG an alternative that addressed this need. Early WYSIWYG:s however didn't utilize SGML or any similar approach, and even today most WYSIWYG:s allow their users to produce unstructured presentationally marked up content.

Another totally different kind of problem that the use of WYSIWYG poses is that people might end up with inappropriate tasks. For example, if a person doing the final layout in a WYSIWYG gets an unstructured text, he or she might not have enough knowledge about subject to apply a sensible styling. Likewise a content editor might not have the proper design skills to do the final styling of the text. These are also problems present when working with content on the Web.

[Taylor(1996)]

2.4.3 Benefits of WYSIWYG

According to TAYLOR the main appeal with WYSIWYG is that it offers the users superior cybernetics, by which he means feedback and control. However this is only to the extent

[5] LATEX and therefore also LYX, the word processor used for this thesis, are using TEX for typesetting

that the feedback and control is authentic, and due to the problems discussed earlier there are plenty of situations where this is not the case. There are however exceptions to this:

> Of course, there are clearly areas of publishing where the enhanced design cybernetics of WYSIWYG are all to the good. This is particularly the case for short, one-time, design-intensive publications where the precise spatial relationship of type and picture elements is critical for aesthetic reasons, such as advertisements, brochures, posters and consumer magazines...
>
> ...A WYSIWYG view is also valuable to designers of information products such as training and procedures manuals, user guides and business forms. In such products, the arrangements of words on a page is a way of reinforcing their meaning, so information designers take care not to set confusing line or page breaks, and may need illustrations to be positioned in a precise relationship both to the accompanying text and to captions.
>
> [Taylor(1996)]

2.5 The tools in practice

2.5.1 Who controls presentation?

In the days when the presentation of web pages was fully controlled internally in the browsers, but a demand for more control over presentation started to grow among web page authors there was an intense discussion on to whom the control over presentation belonged. [Lie and Bos(1999)]

With today's standards based web pages a compromise is reached, where the web pages can have a default presentation defined by using CSS, but the user can override those defaults by using CSS of her own as internal style sheets in the browser.

In today's web, site content and design are often authored by different people, in which case the question arises to what extent the content editor should be able to control presentation. The answer is of course different for different sites, but let's concentrate on rather typical sites such as the one in the case study.

2.5.2 Typical WYSIWYG and CMS usage

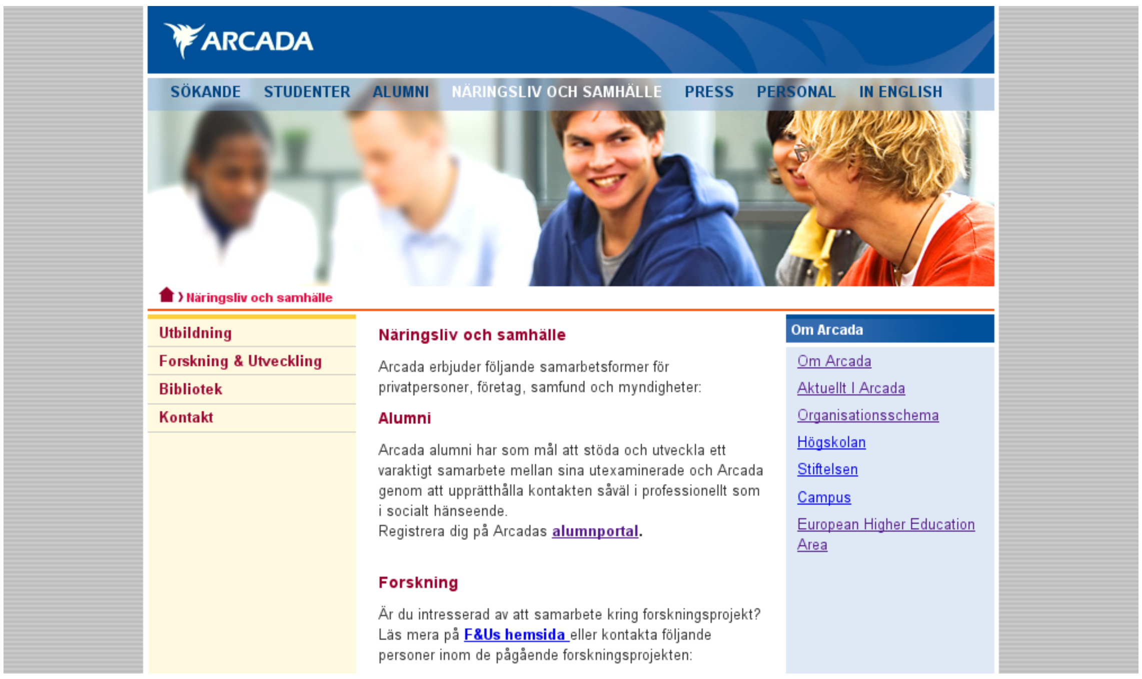
Quite commonly web sites are using a CMS, with HTML/XHTML-templates and CSS designed by a web designer. The CMS lets the user edit certain areas of the content which in the end are inserted into the templates that make up the whole page sent to the browser. So usually the content editor can't touch anything outside of certain areas of the pages.

Often the CMS is equipped with a WYSIWYG, allowing the content editor to see a preview of the page while editing. Quite often this WYSIWYG lets the content editor control also the presentational properties of the areas that are edited. This of course compromises the separation of structure and presentation. The external CSS for the site is out of the WYSIWYG's control, while the WYSIWYG is still able to control certain content areas.

2.5.3 Inconsistency and re-design problems

Not having the control over presentation centralized for the whole site can lead to inconsistent looks. [Meyer(2006), p. 5] This was noted when I worked with a re-design of Arcada polytechnic's web pages. We tried our best to comply with web standards but having hundreds of pages with messy markup it's not an easy task. The pages would have to be cleaned up one by one and then the WYSIWYG (or other kind of editor) provided would have to be changed or configured to not allow such mistakes in the future. Education of the content editors would most probably also be needed, not to mark their texts up according to desired presentation but instead semantically.

Below is a comparison between one page with properly marked up headings and another in which of the headings are just made bold by the content editor. [6]



Näringsliv och samhälle

Arcada erbjuder följande samarbetsformer för privatpersoner, företag, samfund och myndigheter:

Alumni

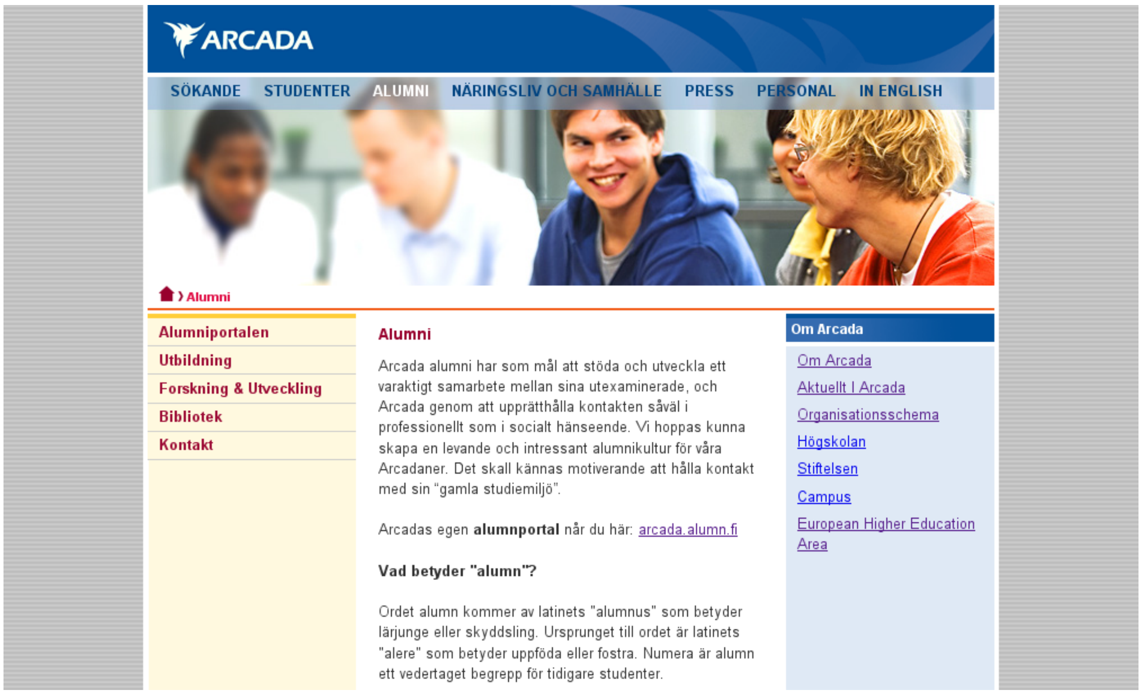
Arcada alumni har som mål att stöda och utveckla ett varaktigt samarbete mellan sina utexaminerade och Arcada genom att upprätthålla kontakten såväl i professionellt som i socialt hänseende.
Registrera dig på Arcadas alumnportal.

Forskning

Är du intresserad av att samarbete kring forskningsprojekt? Läs mera på F&Us hemsida eller kontakta följande personer inom de pågående forskningsprojekten:

Figure 2.2: In this page the headings have been marked up correctly

[6]The screen shots of the pages were taken from a working copy of the site, not the final version.


ARCADA

SÖKANDE STUDENTER ALUMNI NÄRINGSLIV OCH SAMHÄLLE PRESS PERSONAL IN ENGLISH

› Alumni

- Alumniportalen
- Utbildning
- Forskning & Utveckling
- Bibliotek
- Kontakt

Alumni

Arcada alumni har som mål att stöda och utveckla ett varaktigt samarbete mellan sina utexaminerade, och Arcada genom att upprätthålla kontakten såväl i professionellt som i socialt hänseende. Vi hoppas kunna skapa en levande och intressant alumnikultur för våra Arcadaner. Det skall kännas motiverande att hålla kontakt med sin "gamla studiemiljö".

Arcadas egen **alumnportal** når du här: arcada.alumn.fi

Vad betyder "alumn"?

Ordet alumn kommer av latinets "alumnus" som betyder lärjunge eller skyddsling. Ursprunget till ordet är latinets "alere" som betyder uppföda eller fostra. Numera är alumn ett vedertaget begrepp för tidigare studenter.

Om Arcada

- Om Arcada
- Aktuellt I Arcada
- Organisationsschema
- Högskolan
- Stiftelsen
- Campus
- European Higher Education Area

Figure 2.3: In this page some of the headings are not marked up as headings, but only made bold which semantically and even visually speaking doesn't signify them much from ordinary text

It's hard to tell for sure whether the content editor chose to only make some of the headings bold by chance, or with the intent to have them looking in a certain way. Let's pretend that it would have been with the intent to affect their presentational properties, and that affecting this would have been something that we'd allow the content editor to do. In such a case we would have had to provide a tool to control certain parts of the centralized presentation (CSS), a so called Presentation Management System (PMS). [Lombardi(2004)] In this way we could help keeping the look of the site consistent.

However in this case we didn't want to let the content editors control the styling. In fact the control that they did have was even deceptive, since those pages were written before the re-design of the site and looked quite different then. Inconsistently marked up content can even pose bigger problems than we faced with Arcada polytechnic's web site:

> On large sites created by multiple designers and developers, each designer might use different nonstandard tags, making it impossible to gather all the data and reformat it according to a more useful scheme. (Imagine a public library where books were indexed, not by the Dewey Decimal System, but according to the whims of Joe, Mary, and various other cataloguers, each making up their own rules as they went along.)
>
> [Zeldman(2006), p. 86]

Another benefit of a centralized design is performance. Since the centralized CSS file is downloaded only once, only structured content will have to be downloaded repeatedly while browsing around a site. This means a faster and smoother web experience for your visitor, and less server load, bandwidth usage and less traffic costs for your site. [Zeldman(2006), p. 235]

2.5.4 "The Web is not WYSIWYG"

It is not only the possibility of future re-designs that can render the WYSIWYG-concept deceptive. As discussed earlier today's web content is more and more expected to be adaptable for different kinds of media and might also be automatically reused (syndicated) in other web pages.

> Any given Web design will look very different on this variety of devices: clearly, WYSIWYG is dead. Indeed, *looking different is a feature, not a bug*, since an optimal user experience requires adjustments to the characteristics of each device. The more specialized or low-end the device, the stricter the requirements for Web content to morph into something suited for the platform. The only way to make this happen is for designers to give up full control and let the presentation of their pages be determined by an interplay of page specifications and the preference settings and other characteristics of the client device.
>
> [Nielsen(1997)]

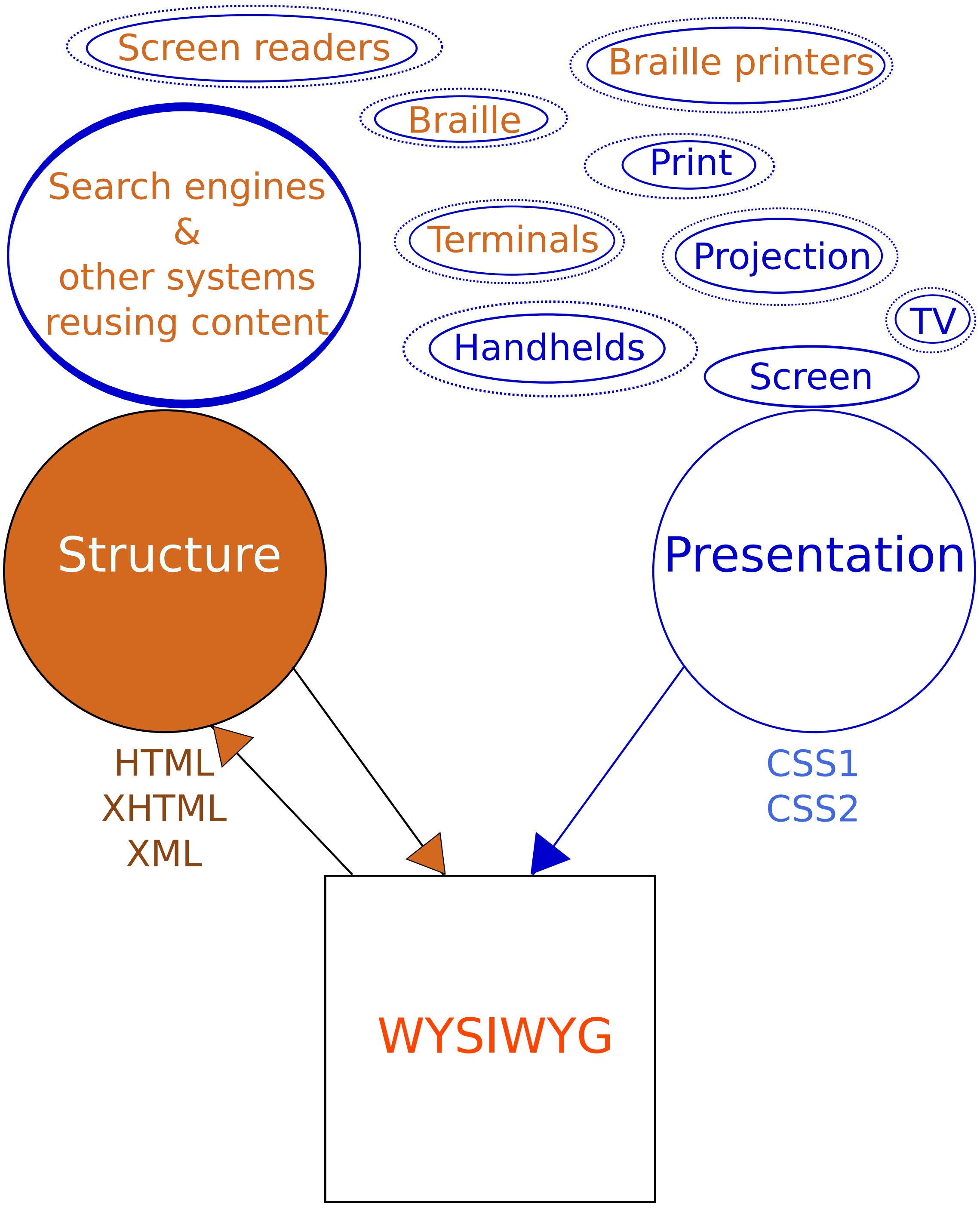


Figure 2.4: A WYSIWYG usually shows a screen-like presentation of the content that is edited, however it only controls document structure (usually XHTML) so any adjustment to the presentation goes into the wrong place (i.e. not the CSS). How the same presentational adjustment will look in a different presentational context or browser gets very unpredictable. This also easily damages the semantic structure of the document which otherwise could ensure a consistent look when adapted to different presentational contexts.

2.5.5 Using a WYSIWYG or something else?

In order to be really honest with the content editor one shouldn't even use a WYSIWYG at all, but instead maybe a WYSIWYM[7], wiki-style editor or something similar such as Textile[8]. A WYSIWYM would show what the different structural elements are, not how they'll eventually be presented. Recently a web based WYSIWYM called WYMeditor[9] was released, and new versions are released frequently. A wiki-style editor lets the user edit plain text with a somewhat simplified markup compared to HTML or XHTML.

But there are those trying to avoid the problems with messy markup while still using a WYSIWYG. The Bitflux Editor is one such example, being one of the few web based WYSIWYG:s that validate the markup created as strict[10] XHTML, that way encouraging use of semantic markup. XHTML 1.0 and 1.1 Strict forbid usage of presentational markup, therefore using strict XHTML and not XHTML 1.0 Transitional is a technical means to try avoiding presentational markup. [Zeldman(2006), p. 149]

Knowing this I asked the lead developer of the Bitflux Editor, CHRISTIAN STOCKER why he after all decided to make it a WYSIWYG:

> ...it's just that our customers ask for WYSIWYG and not WYSIWYM. They mostly do not care about correct semantic usage, but with BXE (Bitflux Editor) you can at least force it somehow...
>
> I know LyX and wrote my diploma thesis with it. It's great and very useful, but I don't see much of a difference between that and a strict BXE. BXE shows what you mean in a manner that it will look like what you will get on the web page. If you reuse the content in a different place (PDA, PDF, whatever), it will of course look differently, but I think the users can abstract that.
>
> [Stocker(2005)]

It seems like having a WYSIWYG in a CMS is something that people expect, probably due its familiar word-processor like interface and its seemingly good cybernetics[11]. Since WYSIWYG:s are commonly used, it's worth exploring further how well a WYSIWYG producing strict XHTML will do concerning separation of structure and presentation. This will be done with the Bitflux Editor in the case study of this thesis.

[7]What You See Is What You Mean, a concept founded for the very word processor with which this thesis was written, LyX `http://www.lyx.org/`

[8]`http://textism.com/tools/textile/`

[9]`http://www.wymeditor.org/`

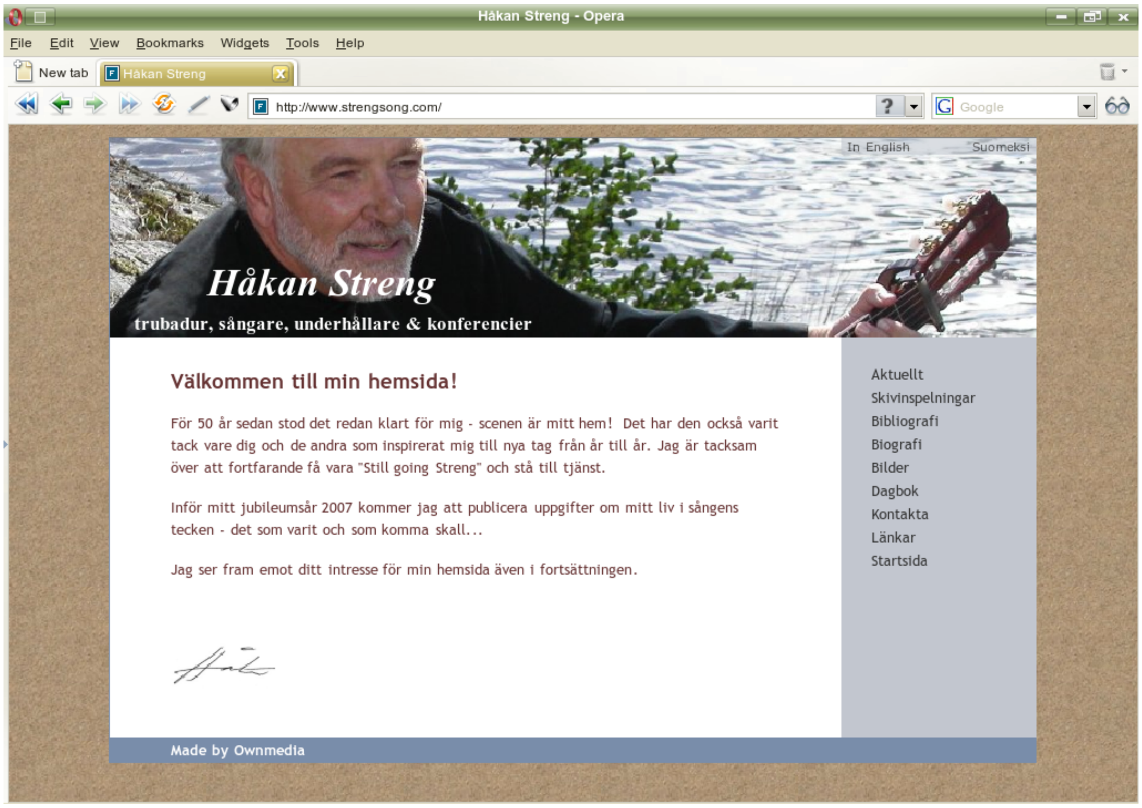
[10]The Bitflux Editor doesn't validate the markup precisely according to the XHTML 1.0 Strict schema, but in practice no invalid XHTML 1.0 Strict markup was found in the case study. The RelaxNG schema that it uses by default for XHTML-validation is found here: `http://svn.bitflux.ch/repos/public/fluxcms/trunk/inc/bx/doctypes/schemas/xhtml/xhtml-basic.rng`

[11]See 2.4.3 on page 20

2.6 The case

2.6.1 The site

The site in the case study, the troubadour Håkan Streng's web site, is a quite ordinary site consisting of mostly static pages with mostly text content that the content editor can edit. It is in that sense quite similar to Arcada polytechnic's site discussed in 2.5.3 on page 22, but has much fewer pages.



Välkommen till min hemsida!

För 50 år sedan stod det redan klart för mig - scenen är mitt hem! Det har den också varit tack vare dig och de andra som inspirerat mig till nya tag från år till år. Jag är tacksam över att fortfarande få vara "Still going Streng" och stå till tjänst.

Inför mitt jubileumsår 2007 kommer jag att publicera uppgifter om mitt liv i sångens tecken - det som varit och som komma skall...

Jag ser fram emot ditt intresse för min hemsida även i fortsättningen.

[illegible]

Figure 2.5: The front page of the site, the first level of the navigation three is visible in the right hand column. Only the Swedish version of the site is studied, since it's the only language in which all of the pages are available.

The whole navigation three of the site translated into English:

- Current events
- Records
  - *CD*
- Bibliography
- Biography
  - Personal information
- *Gallery*
- *Diary*
- *Contact*
- Links
- Front page

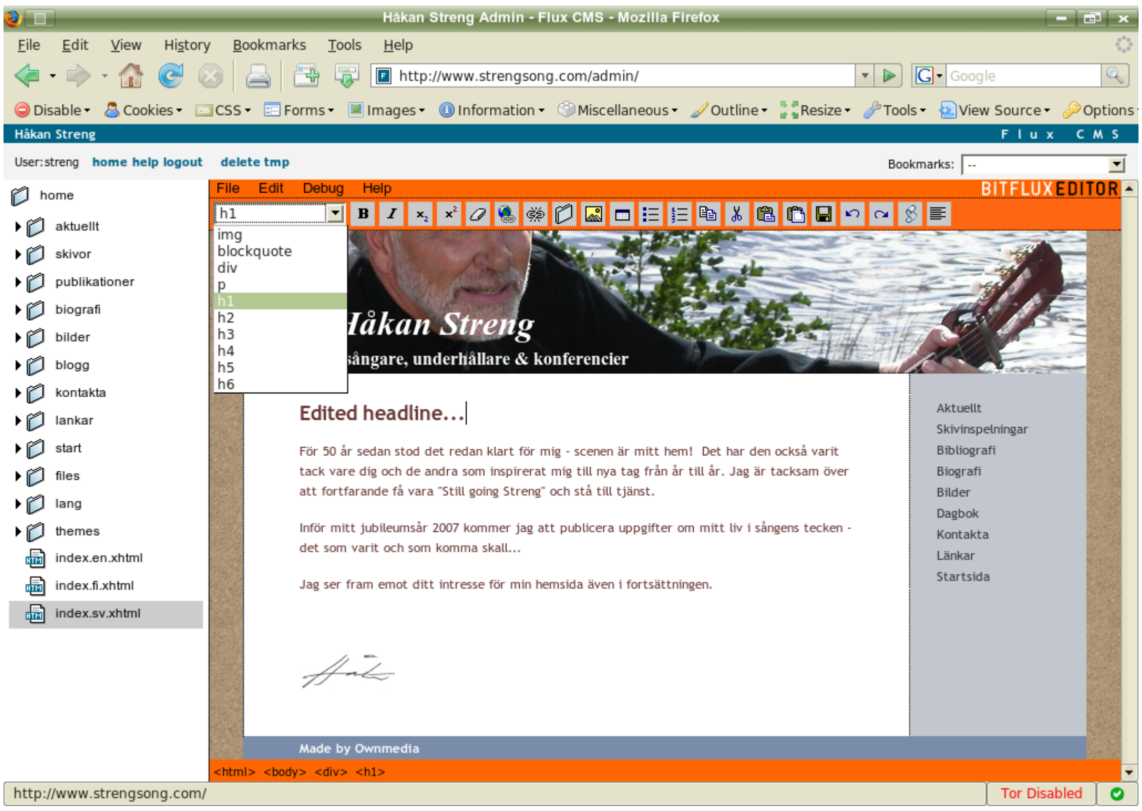
The pages *CD*, *Gallery*, *Diary* and *Contact* are not a part of the case study, since they have not been edited by the content editor using the Bitflux Editor. *CD* uses special formatting, not achievable through the Bitflux Editor, *Gallery* is controlled through a gallery feature in Flux CMS and not a WYSIWYG, *Diary* is driven by a blogging tool within Flux CMS (not using the Bitflux Editor) and *Contact* is a contact form, not formatted by the content editor.

The rest of the pages are quite ordinary static web pages, edited by the content editor through the Bitflux Editor. These pages are the ones being examined in the case study.

2.6.2 Bitflux Editor and Flux CMS

The pages of the site are managed through Flux CMS[12], and the pages are edited with the WYSIWYG Bitflux Editor[13] inside of the CMS.

[12]http://www.flux-cms.org/
[13]http://bitfluxeditor.org/



Edited headline...

För 50 år sedan stod det redan klart för mig - scenen är mitt hem! Det har den också varit tack vare dig och de andra som inspirerat mig till nya tag från år till år. Jag är tacksam över att fortfarande få vara "Still going Streng" och stå till tjänst.

Inför mitt jubileumsår 2007 kommer jag att publicera uppgifter om mitt liv i sångens tecken - det som varit och som komma skall...

Jag ser fram emot ditt intresse för min hemsida även i fortsättningen.

Figure 2.6: The Bitflux Editor covers the area with the top border with the text “BITFLUX EDITOR”, the rest of the web page with a top border reading “Flux CMS” is Flux CMS, that embeds the Bitflux Editor. The has been edited and the pop-up menu enables the user to change element type. (Refer to 2.3.4 on page 18 for explanations on some of the element types.)

A stable development snap-shoot of Flux CMS, version 1.5.0-dev/r8109 is used, since much development has happened since the last official release at the time of this writing. However as far as this thesis is concerned, the nature of the CMS used doesn’t matter as long as it doesn’t pose problems concerning the separation of structure and presentation. Flux CMS handles this perfectly and is therefore not discussed any further.

Bitflux Editor version 1.1.0-dev/200609280330/r1517 was included with the Flux CMS version in question and was therefore used to edit the pages. In the case study the Bitflux Editor produced code that validates as XHTML 1.0 Strict, which doesn’t allow markup with explicitly presentational properties. Exceptions are the tags <*b*> and <*i*> that represent *bold* and *italic* styling and lack semantic meaning. The Bitflux Editor’s *bold* and *italic* buttons[14] were by default configured to produce <*b*> and <*i*>, to make the Bitflux Editor only produce semantic markup it was reconfigured to produce <*strong*> and <*em*> instead, which have the semantic meanings *strong emphasize* and *emphasize*, but still the same default styling as <*b*> and <*i*>. While it’s slightly deceptive to make the *bold* and *italic* buttons produce semantic markup, it’s however common practice in many

[14]the B and I buttons right from the pop-up menu in figure 2.6

WYSIWYG:s and usually appropriate when marking up ordinary text.

The Bitflux Editor allows one to configure schemas defining exactly what kinds of elements are allowed where. However this was not used to prevent presentational usage of markup, since we want to figure out in what ways the content editor uses valid XHTML 1.0 Strict markup for presentational purposes.

2.7 Earlier studies

2.7.1 XML WYSIWYG-editors

The Bitflux Editor is actually a general XML-editor, even though only used for XHTML in the case study. There has been a bit of research about XML editors with WYSIWYG capabilities, mostly with non web based ones. Many of them have capabilities to view multiple presentations, which makes sense since XML documents (including XHTML documents) should be structured data being presentable in multiple ways. Some are also in practice more like that what has been referred to as WYSIWYM, though being called WYSIWYG, probably because WYSIWYM is a comparably unknown word. However none of the research explored the problem with the user using markup in a presentational manner. [Schrage(2004)] [Ho(2005)]

2.7.2 A cleanup approach

Research was also found on trying to clean up messy code produced by WYSIWYG:s afterwards. Even though this might be an interesting approach, especially when doing a redesign having a lot of messy content from before, it is not focused on messy code produced solely by the content editor's presentational usage of markup. [Spiesser and Kitchen(2004)]

2.8 Method

2.8.1 Noting inappropriately used markup

Since no applicable research method was found from earlier research on similar subjects, a new one had to be developed to suite our needs:

The content editor got brief instructions on how to mark the content up semantically. The headings were taken as an example since they are the most commonly used element that easily could be wrongly used. Instructions were given to assign headings starting with first level headings, then second level and so on.

In the case study we'll go through the ways in which different pages in the case, Håkan Streng's web site, failed to keep separation of structure and presentation intact. In practice this means that we'll look at in which ways the content editor after all has used the WYSIWYG to control presentational aspects of the pages instead of structural.

Markup that is inappropriate from a structural point of view will be noted. Due to a lot of markup not producing any visual result, therefore not being able to be called presentational, the results are sorted into markup producing a visible result and markup not producing any visible result. Markup not producing any visible result is likely to have been added by mistake, and therefore can tell us something about what kinds of unintentional mistakes that the content editor has made. This information can be useful when analyzing the inappropriately used markup that did cause visual results. Not all of it is necessarily to be considered presentationally used, some of it may have been added by mistake as well.

2.8.2 Tools

The pages' presentational outcome is only checked in the Mozilla Firefox[15] browser, since that's the only browser in which the Bitflux Editor works. The Bitflux Editor's WYSIWYG view utilizes the web browser's rendering engine and therefore shows the pages exactly like Firefox would with the readily edited page. Possible differing presentational results in other browsers are highly unlikely to have been intentionally made by the editor, since they wouldn't have been shown while editing.

Since the pages were edited using different versions of Firefox they were checked with Firefox 1.0.8, 1.5.0.12 and 2.0.0.4 to make sure that no rendering differences were to be found. In the case study the code and presentation is studied using the Firefox extension Firebug[16] 1.05 in Firefox 2.0.0.4, as shown in the figures in the case study. The operating system used is Kubuntu[17] Linux 7.04 Feisty Fawn.

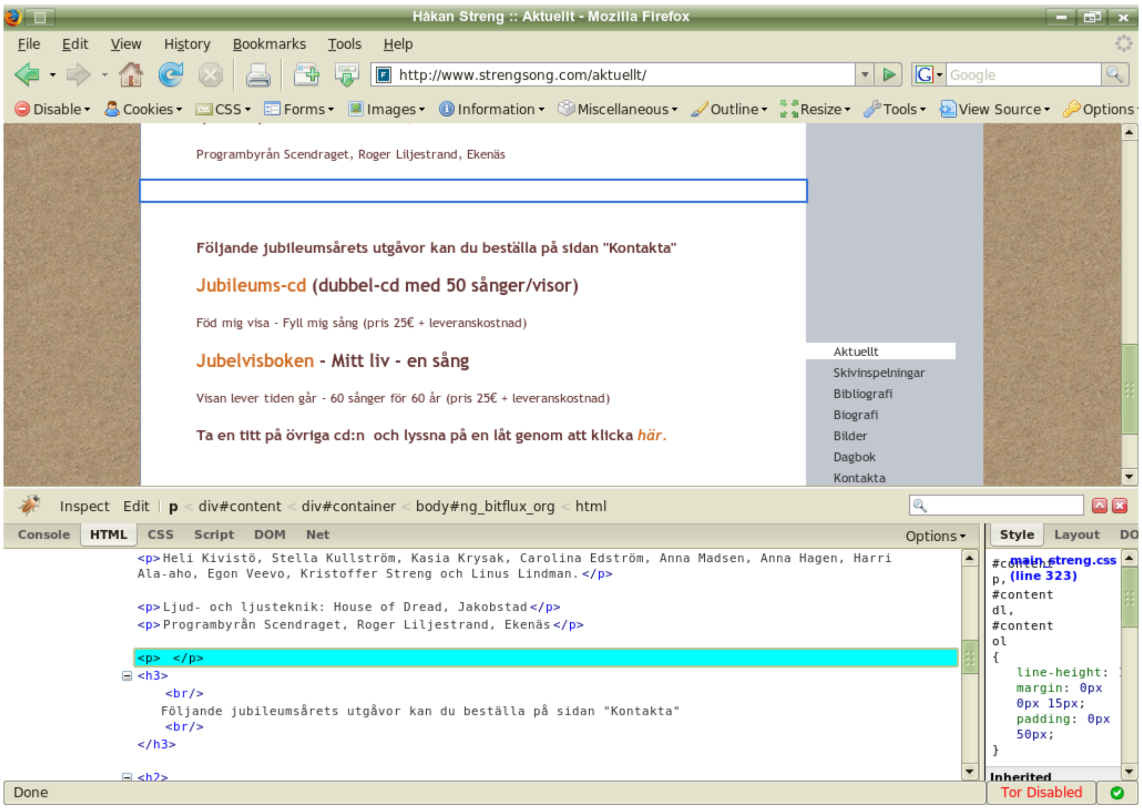
[15]http://www.mozilla.com/firefox/
[16]http://www.getfirebug.com/
[17]http://www.kubuntu.org/

3 The case study

3.1 Inappropriately used markup

3.1.1 Empty paragraphs

In five instances empty paragraphs were used, making empty space. In figure 3.1 and in one other instance only one empty paragraph was used, while as in two of the other instances two and three empty paragraphs were used creating even more empty space. In figure 3.2 an empty heading was used in combination with an empty paragraph.



Håkan Streng :: Aktuellt - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://www.strengsong.com/aktuellt/
Google
Disable Cookies CSS Forms Images Information Miscellaneous Outline Resize Tools View Source Options
Aktuellt
Skivinspelningar
Bibliografi
Biografi
Bilder
Dagbok
Kontakta
Inspect Edit p < div#content < div#container < body#ng_bitflux_org < html
Console HTML CSS Script DOM Net Options
Style Layout DO
Inherited
Done
Tor Disabled

Programbyrån Scendraget, Roger Liljestrand, Ekenäs

Följande jubileumsårets utgåvor kan du beställa på sidan "Kontakta"

Jubileums-cd (dubbel-cd med 50 sånger/visor)

Föd mig visa - Fyll mig sång (pris 25€ + leveranskostnad)

Jubelvisboken - Mitt liv - en sång

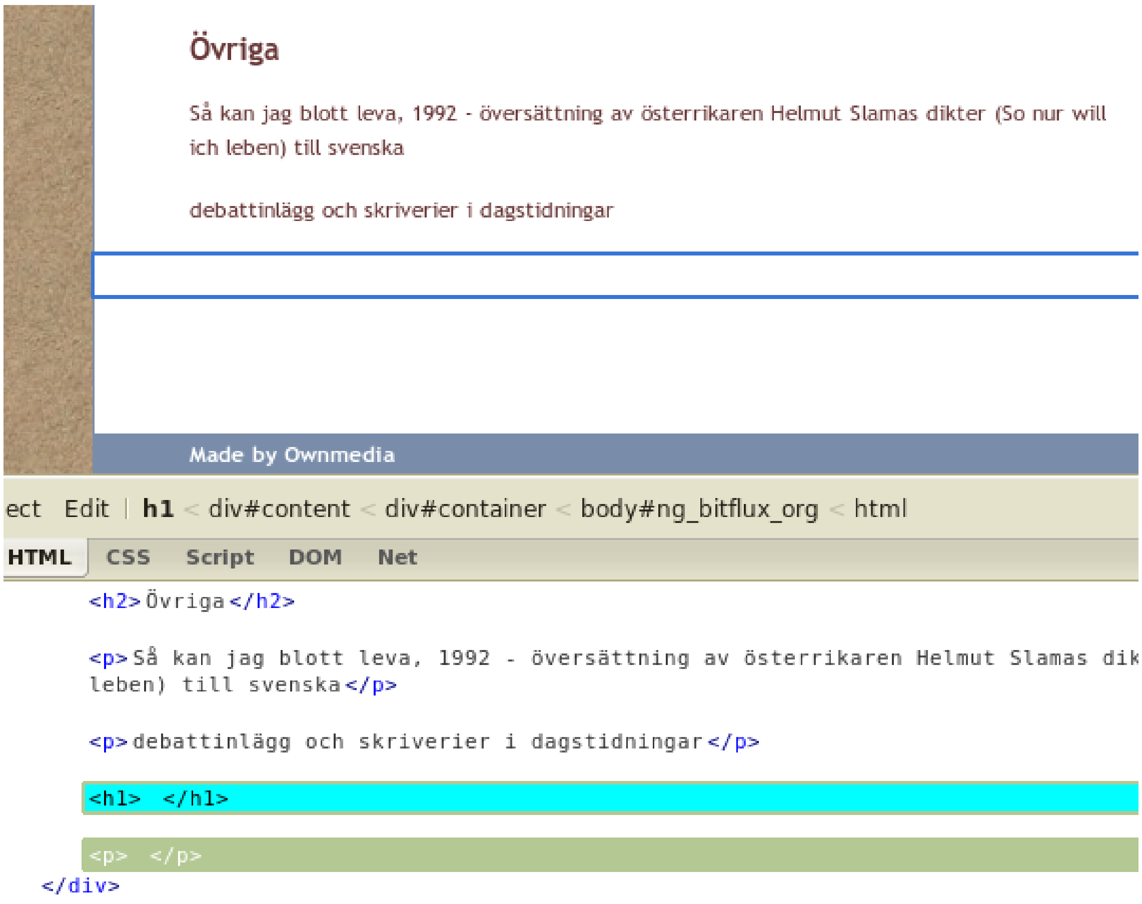
Visan lever tiden går - 60 sånger för 60 år (pris 25€ + leveranskostnad)

Ta en titt på övriga cd:n och lyssna på en låt genom att klicka här.

```
<p>Heli Kivistö, Stella Kullström, Kasia Krysak, Carolina Edström, Anna Madsen, Anna Hagen, Harri
Ala-aho, Egon Veevo, Kristoffer Streng och Linus Lindman.</p>

<p>Ljud- och ljusteknik: House of Dread, Jakobstad</p>
<p>Programbyrån Scendraget, Roger Liljestrand, Ekenäs</p>

<p> </p>
<h3>
    <br/>
   Följande jubileumsårets utgåvor kan du beställa på sidan "Kontakta"
    <br/>
</h3>

<h2>
```

```
main_streng.css (line 323)
#content
p,
#content
dl,
#content
ol
{
   line-height:
   margin: 0px
   0px 15px;
   padding: 0px
   50px;
}
```

Figure 3.1: On the upper half of the picture you see a box marking the empty paragraph as displayed on the web page, below that you see a part of the XHTML-code where the code making up the empty paragraph is marked with a line.


Övriga

Så kan jag blott leva, 1992 - översättning av österrikaren Helmut Slamas dikter (So nur will ich leben) till svenska

debattinlägg och skriverier i dagstidningar

Made by Ownmedia

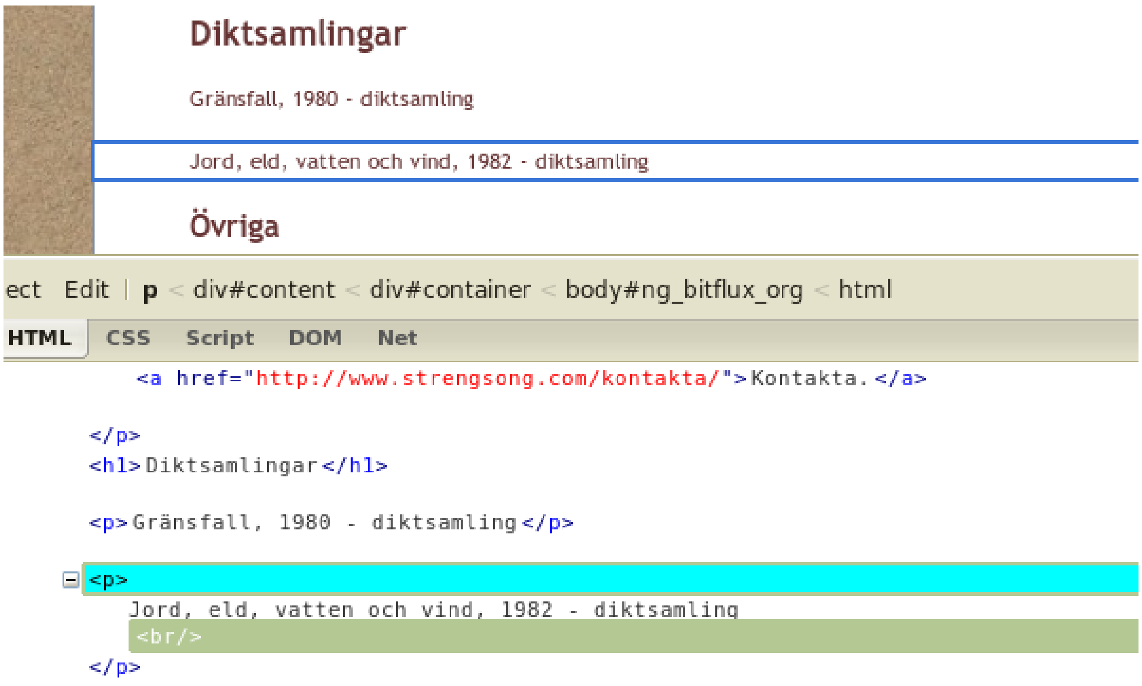
ect Edit | h1 < div#content < div#container < body#ng_bitflux_org < html
HTML CSS Script DOM Net

```
      <h2>Övriga</h2>

      <p>Så kan jag blott leva, 1992 - översättning av österrikaren Helmut Slamas dik
      leben) till svenska</p>

      <p>debattinlägg och skriverier i dagstidningar</p>

      <h1> </h1>

      <p> </p>
   </div>
```

Figure 3.2: An empty heading in combination with an empty paragraph, making empty space

3.1.2 Inappropriate use of line breaks

In nine instances line breaks were inappropriately used. In figure 3.3 and two other instances a line break was put at the end of a paragraph, while in figure 3.4 and one other similar case a line break was put at the end of an unordered list, none of them causing any visual result. In two of the instances the line breaks were put at the beginning and in one at the end of a heading, both kinds occur in figure 3.5, when put at the beginning of a heading making space above the heading's text, but when put after the heading making no visual result. In figure 3.6 two line breaks were put above a an image, adding extra space.



Diktsamlingar

Gränsfall, 1980 - diktsamling

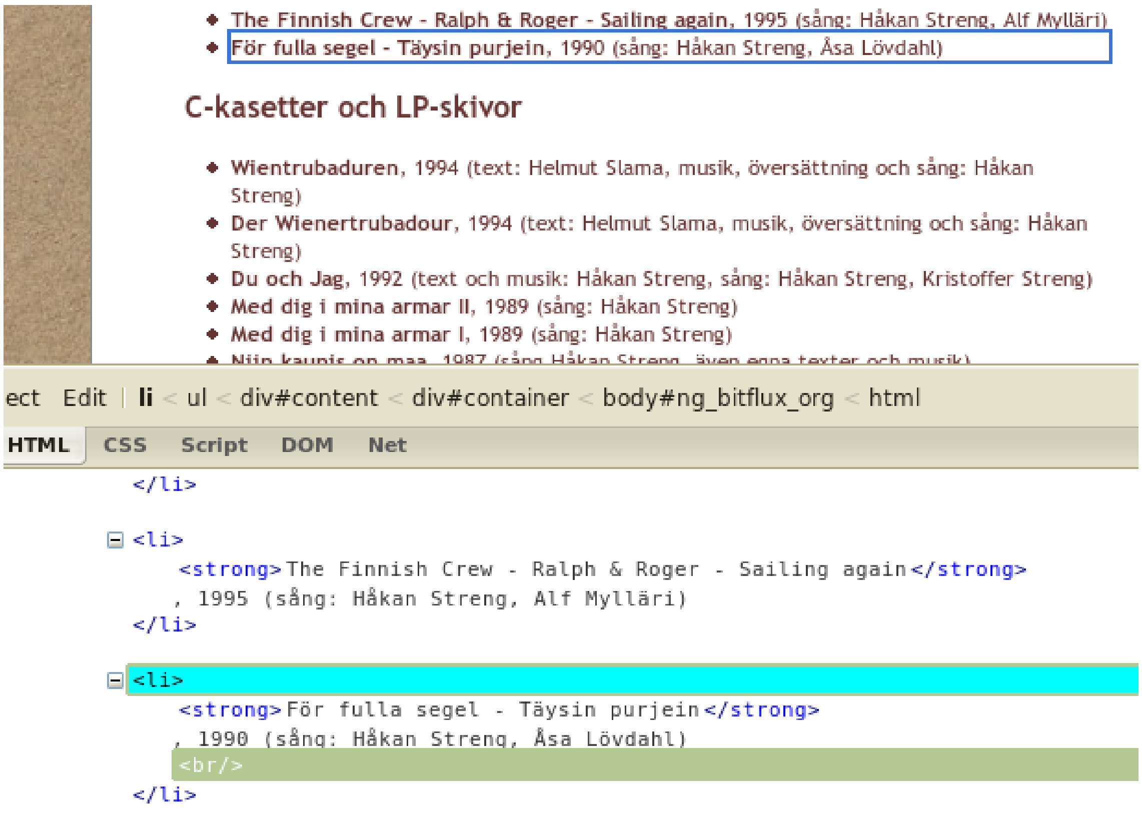
Jord, eld, vatten och vind, 1982 - diktsamling

Övriga

ect Edit | p < div#content < div#container < body#ng_bitflux_org < html
HTML CSS Script DOM Net

```
    <a href="http://www.strengsong.com/kontakta/">Kontakta.</a>

</p>
<h1>Diktsamlingar</h1>

<p>Gränsfall, 1980 - diktsamling</p>

<p>
    Jord, eld, vatten och vind, 1982 - diktsamling
    <br/>
</p>
```

Figure 3.3: A line break put at the end of a paragraph, causing no visual result



- **The Finnish Crew - Ralph & Roger - Sailing again**, 1995 (sång: Håkan Streng, Alf Mylläri)
- **För fulla segel - Täysin purjein**, 1990 (sång: Håkan Streng, Åsa Lövdahl)

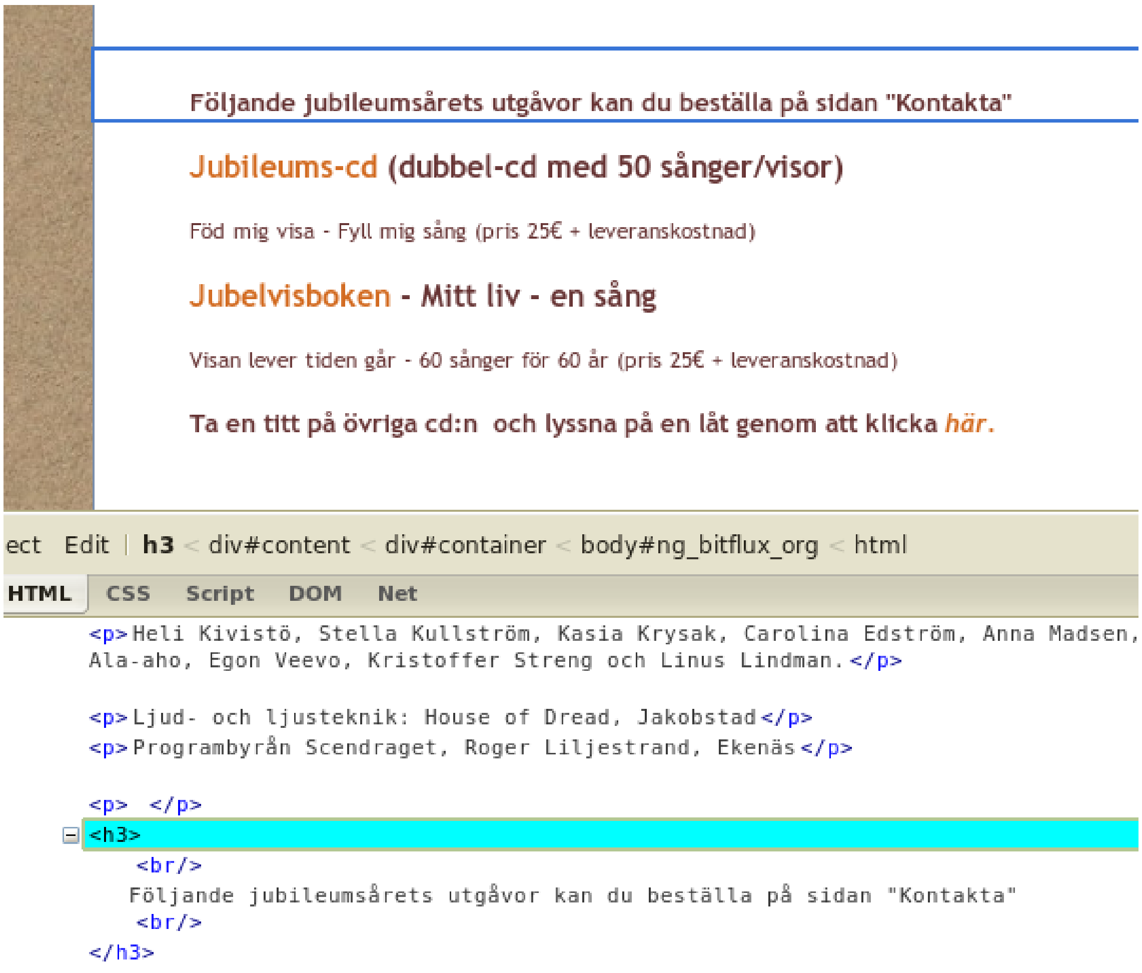
C-kasetter och LP-skivor

- **Wientrubaduren**, 1994 (text: Helmut Slama, musik, översättning och sång: Håkan Streng)
- **Der Wienertrubadour**, 1994 (text: Helmut Slama, musik, översättning och sång: Håkan Streng)
- **Du och Jag**, 1992 (text och musik: Håkan Streng, sång: Håkan Streng, Kristoffer Streng)
- **Med dig i mina armar II**, 1989 (sång: Håkan Streng)
- **Med dig i mina armar I**, 1989 (sång: Håkan Streng)

ect Edit | li < ul < div#content < div#container < body#ng_bitflux_org < html
HTML CSS Script DOM Net

```
</li>

<li>
    <strong>The Finnish Crew - Ralph & Roger - Sailing again</strong>
    , 1995 (sång: Håkan Streng, Alf Mylläri)
</li>

<li>
    <strong>För fulla segel - Täysin purjein</strong>
    , 1990 (sång: Håkan Streng, Åsa Lövdahl)
    <br/>
</li>
```

Figure 3.4: Line break put at the end of a list item, causing no visual result


ect Edit | h3 < div#content < div#container < body#ng_bitflux_org < html
HTML CSS Script DOM Net

Följande jubileumsårets utgåvor kan du beställa på sidan "Kontakta"

Jubileums-cd (dubbel-cd med 50 sånger/visor)

Föd mig visa - Fyll mig sång (pris 25€ + leveranskostnad)

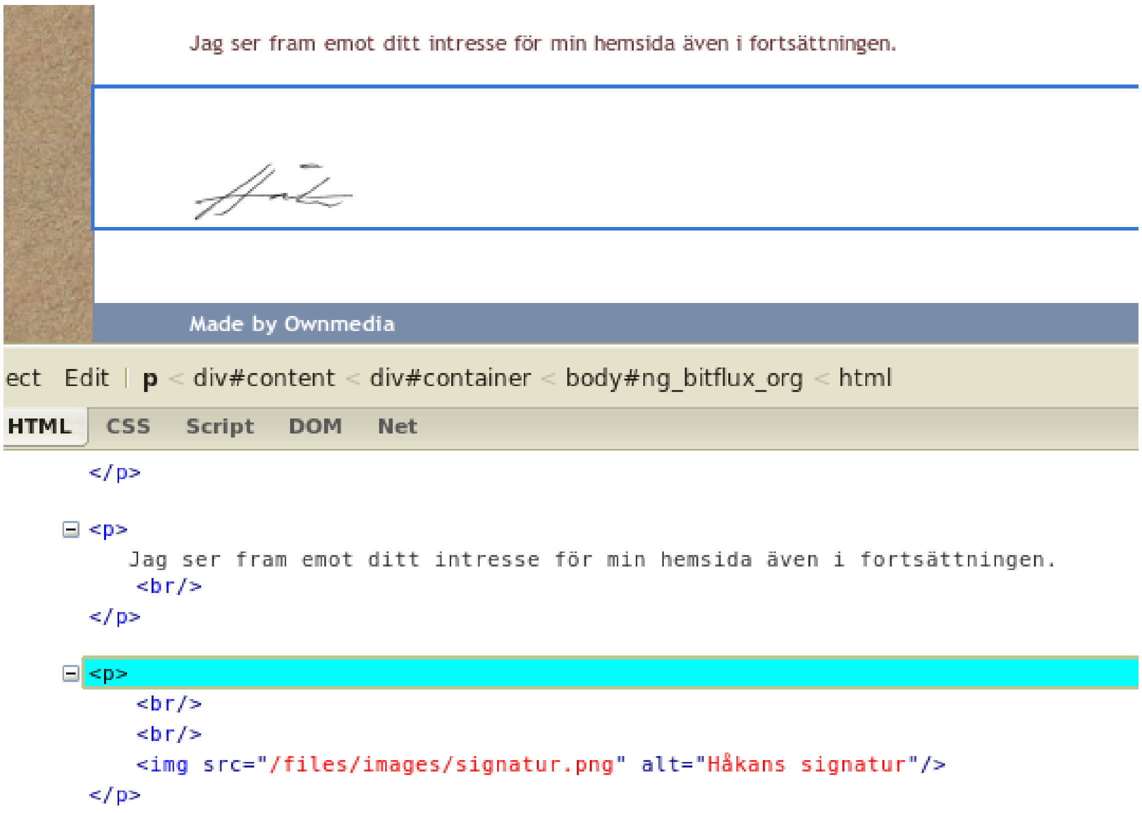
Jubelvisboken - Mitt liv - en sång

Visan lever tiden går - 60 sånger för 60 år (pris 25€ + leveranskostnad)

Ta en titt på övriga cd:n och lyssna på en låt genom att klicka *här*.

```
<p>Heli Kivistö, Stella Kullström, Kasia Krysak, Carolina Edström, Anna Madsen,
Ala-aho, Egon Veevo, Kristoffer Streng och Linus Lindman.</p>

<p>Ljud- och ljusteknik: House of Dread, Jakobstad</p>
<p>Programbyrån Scendraget, Roger Liljestrand, Ekenäs</p>

<p> </p>
<h3>
   <br/>
  Följande jubileumsårets utgåvor kan du beställa på sidan "Kontakta"
   <br/>
</h3>
```

Figure 3.5: Line breaks added in the beginning and the end of the heading, adding more space above it, but none underneath it


Jag ser fram emot ditt intresse för min hemsida även i fortsättningen.

Made by Ownmedia

ect Edit | p < div#content < div#container < body#ng_bitflux_org < html
HTML CSS Script DOM Net

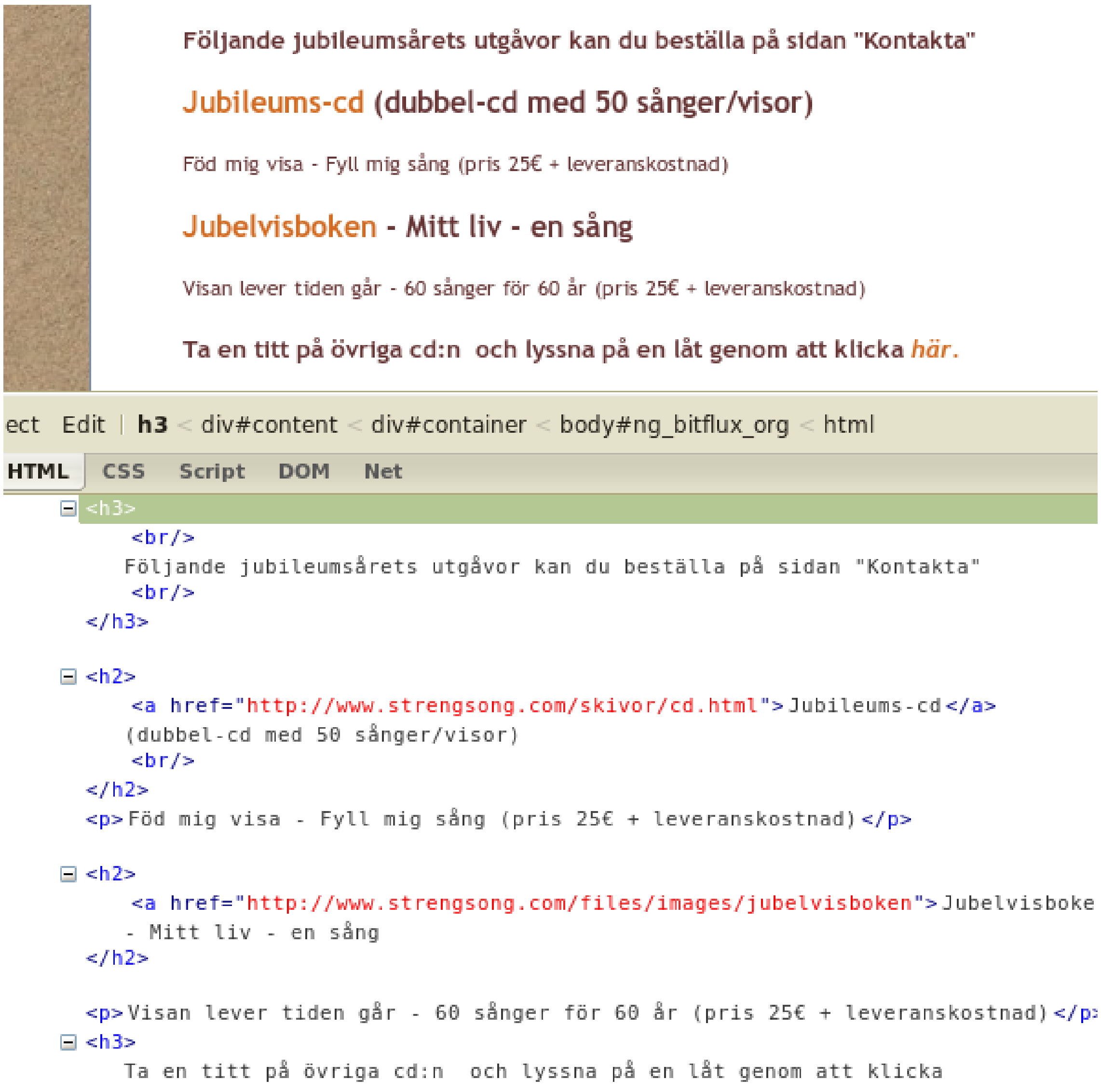
```
</p>

<p>
    Jag ser fram emot ditt intresse för min hemsida även i fortsättningen.
    <br/>
</p>

<p>
    <br/>
    <br/>
    <img src="/files/images/signatur.png" alt="Håkans signatur"/>
</p>
```

Figure 3.6: Two line breaks adding space above the image with the signature

3.1.3 Inappropriate use of headings

In four instances the choice of heading depth was semantically incorrect, all are shown in figure 3.7.

In one additional instance, as shown in figure 3.2, an empty heading was used, making empty space.


Följande jubileumsårets utgåvor kan du beställa på sidan "Kontakta"

Jubileums-cd (dubbel-cd med 50 sånger/visor)

Föd mig visa - Fyll mig sång (pris 25€ + leveranskostnad)

Jubelvisboken - Mitt liv - en sång

Visan lever tiden går - 60 sånger för 60 år (pris 25€ + leveranskostnad)

Ta en titt på övriga cd:n och lyssna på en låt genom att klicka *här*.

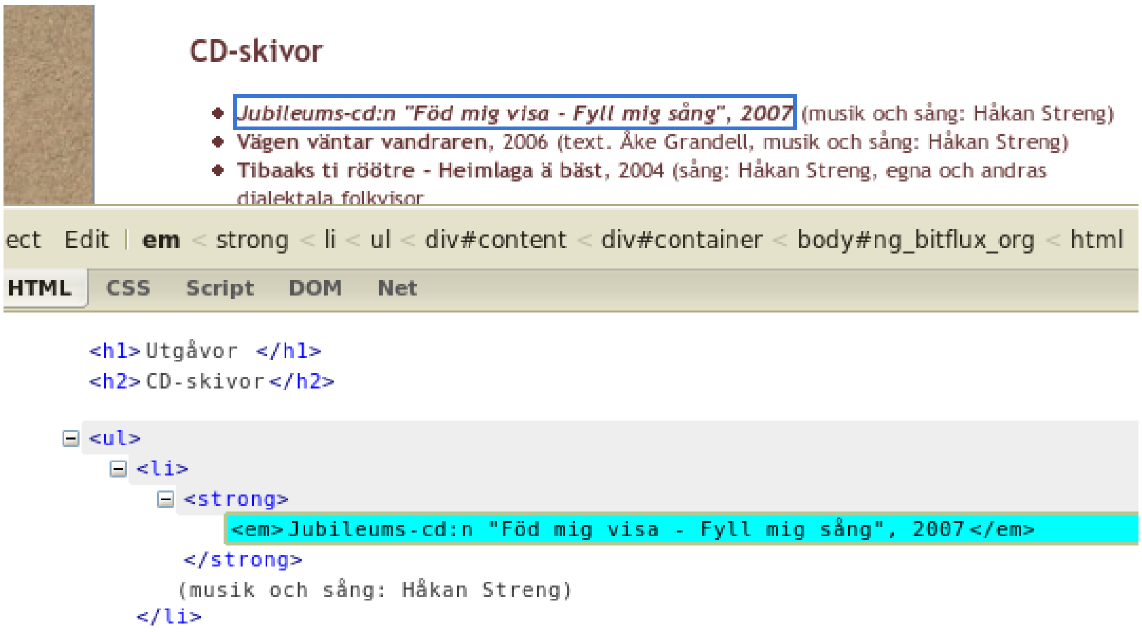
ect Edit | h3 < div#content < div#container < body#ng_bitflux_org < html
HTML CSS Script DOM Net

```
<h3>
    <br/>
   Följande jubileumsårets utgåvor kan du beställa på sidan "Kontakta"
    <br/>
</h3>

<h2>
    <a href="http://www.strengsong.com/skivor/cd.html">Jubileums-cd</a>
   (dubbel-cd med 50 sånger/visor)
    <br/>
</h2>
<p>Föd mig visa - Fyll mig sång (pris 25€ + leveranskostnad)</p>

<h2>
    <a href="http://www.strengsong.com/files/images/jubelvisboken">Jubelvisboke
   - Mitt liv - en sång
</h2>

<p>Visan lever tiden går - 60 sånger för 60 år (pris 25€ + leveranskostnad)</p
<h3>
   Ta en titt på övriga cd:n  och lyssna på en låt genom att klicka
```

Figure 3.7: The first third level sub-heading is obviously inappropriately used since it's directly followed by a second level sub-heading instead of some content which it would represent. The page also ends with a third level heading, followed by no content at all. Also the two second level headings in the middle are inappropriately used because they actually contain text that is to it's characteristics more like ordinary page content, even though that may not be as obvious as with the previous examples.

3.1.4 Emphasize and strong emphasize combined

In two instances emphasize and strong emphasize were combined producing a combination of bold and italic text. In figure 3.8 this is achieved while as in figure 3.9 strong emphasize doesn't have any visual effect, since the fact that the text was within a heading made it bold by default. The fact that the affected word is a link makes it have a slightly different color.

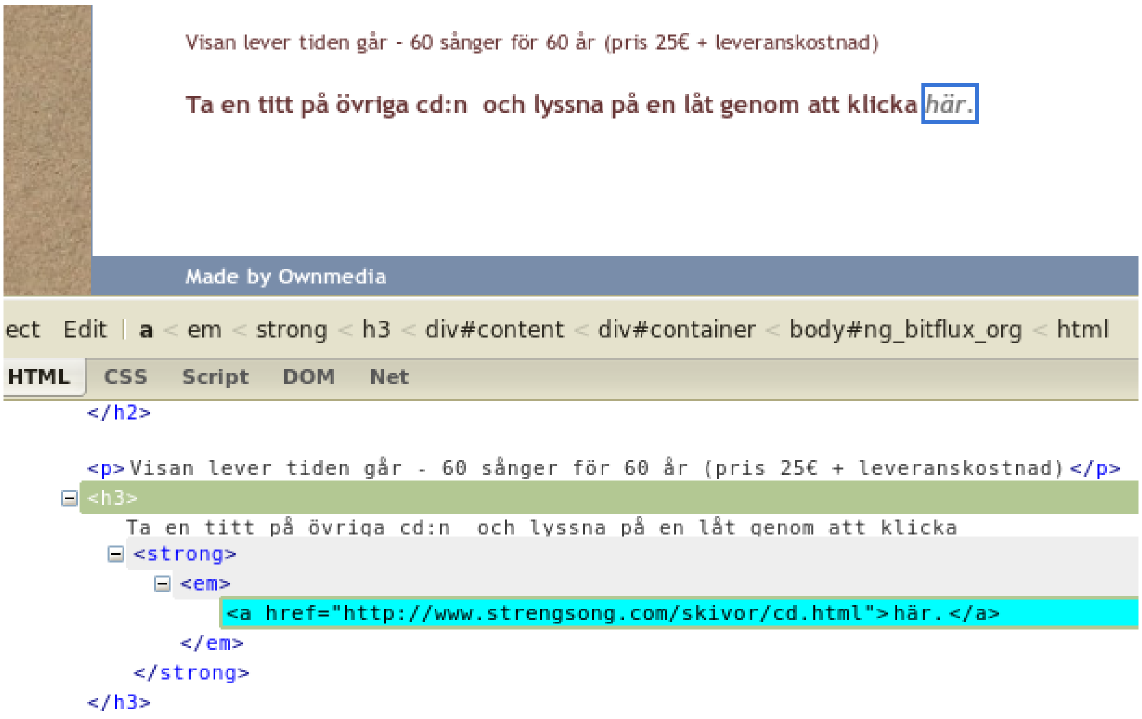

CD-skivor

- *Jubileums-cd:n "Föd mig visa - Fyll mig sång", 2007* (musik och sång: Håkan Streng)
- Vägen väntar vandraren, 2006 (text. Åke Grandell, musik och sång: Håkan Streng)
- Tibaaks ti röötre - Heimlaga ä bäst, 2004 (sång: Håkan Streng, egna och andras dialektala folkvisor

ect Edit | em < strong < li < ul < div#content < div#container < body#ng_bitflux_org < html
HTML CSS Script DOM Net

```
    <h1>Utgåvor </h1>
    <h2>CD-skivor</h2>

  <ul>
    <li>
      <strong>
        <em>Jubileums-cd:n "Föd mig visa - Fyll mig sång", 2007</em>
      </strong>
      (musik och sång: Håkan Streng)
    </li>
```

Figure 3.8: Emphasize and strong emphasize used in combination causing a combination of both italic and bold text styling


Visan lever tiden går - 60 sånger för 60 år (pris 25€ + leveranskostnad)

Ta en titt på övriga cd:n och lyssna på en låt genom att klicka *här.*

Made by Ownmedia

ect Edit | a < em < strong < h3 < div#content < div#container < body#ng_bitflux_org < html
HTML CSS Script DOM Net

```
    </h2>

    <p>Visan lever tiden går - 60 sånger för 60 år (pris 25€ + leveranskostnad)</p>
  <h3>
      Ta en titt på övriga cd:n  och lyssna på en låt genom att klicka
    <strong>
      <em>
        <a href="http://www.strengsong.com/skivor/cd.html">här.</a>
      </em>
    </strong>
  </h3>
```

Figure 3.9: A linked word in a heading, made italic because of emphasize usage, while the use of strong emphasize has no visual effect since the fact that it's a part of a heading makes it bold already by default

3.2 Summary

Table 3.1: Number of instances with inappropriate markup, the amount of those instances causing a visible as well as invisible results specified separately

	Instances	Invisible	Visible
Paragraphs	5	0	5
Line breaks	9	7	2
Headings	5	0	5
Emphasize	2	9	2
Strong emphasize	2	1	1
Total	23	8	15

4 Analysis

4.1 Overview

Some of the inappropriately used markup may have been added by mistake, while as most is very likely to have been added deliberately to affect the presentational outcome.

4.2 Unclear cases

4.2.1 Paragraphs and one heading

The empty paragraphs could easily have been added by mistake but are in most cases quite likely to have been made deliberately to make up space. Figure 3.2 on page 33 on the other hand looks like it could have contained content, but eventually would have been left empty. First there is an empty heading followed by a paragraph, just as there would be if the empty heading would have been a properly used one. This makes it look like it could have been left like that by mistake.

4.3 Clear cases

4.3.1 Line breaks

The use of line breaks at the end of list items, headings and in one case a paragraph did not cause any visual result as opposed to when put in the beginning of a heading. However the Bitflux Editor requires its user to hold down the shift-button while pushing enter in order to achieve a line break, which makes it unlikely that any of the line breaks would have been added by mistake, even though not giving the supposedly expected visual result in most cases.

4.3.2 Headings

Figure 3.7 on page 37 clearly shows that there is no reason to suspect that the inappropriately used headings (except for the one mentioned in 4.2.1) would have been made by mistake. Figure 3.7 also shows that they gave a clearly visible visual result.

4.3.3 Emphasize and strong emphasize

There is no reason to suspect that emphasize and strong emphasize would have been used for any other reason than a presentational. This also applies to the usage of strong emphasize in figure 3.9 on page 38 even though it does not happen to produce any visible result. The content editor is unlikely to have known that strong emphasize wouldn't have any visual effect within a heading, and the use of a combination of emphasize and strong emphasize doesn't semantically make sense either.

4.4 Conclusion

All inappropriate usage of markup except for the usage of empty paragraphs and in one case an empty heading is more or less clearly deliberate, no matter whether it ended up producing a visible result or not. Exceptionally clear is the intent behind the misuse of headings to achieve certain text styling.

Even though the intent behind some of the cases remains a bit uncertain, it can be concluded that in some cases the content editor's ambition to control presentation is clearly behind the usage of inappropriate markup.

Apparently even a WYSIWYG intended for structural content can mislead its user into marking up the content according to presentation instead of semantic structure. The WYSIWYG-concept in itself becomes misleading in such a context. In the case study the content editor even received a briefing in semantic structuring, as explained in 2.8.1 on page 30. Even though these guidelines were followed to some extent, they were occasionally ignored when needed in order to achieve a certain presentational effect.

5 Summary and discussion

5.1 Evaluation

The central argument[1] that “separation of structure and presentation cannot be maintained purely by technical means, while using a WYSIWYG such as the Bitflux Editor” was proven. Even though the goal to prove that was attained, the method would have been more accurate if the content editor’s intent with each action would have been monitored. In theory even any inappropriately used markup could have been added by mistake, however in practice this is very unlikely since there seems to be a clear pattern of seeking presentational effects behind most of it.

The assumption that there would be a connection between the visibility of the results and the content editor’s intent however turned out to be false.

5.2 Recommendations

5.2.1 Abandon WYSIWYG for web content

Given that the WYSIWYG-concept initially was invented for print production, with one single presentational outcome, it’s natural that it might not be fit for the Web. As discussed in 2.4.3 on page 20, WYSIWYG:s sometimes do have their benefits due to superior cybernetics[2]. With (HTML/XHTML-based) web content this is however not the case, since the cybernetics remain false due to the multiple possible presentational outcomes. This means that the user might get tricked into thinking that he or she has ultimate control over how the content will look when presented. This ultimate control is impossible since the content will look slightly different in different ordinary browsers and very different in entirely different presentational media, such as mobile devices.

[1]See 1.3 on page 10
[2]feedback and control

Non-CMS contexts

Some non-HTML/XHTML page elements or file formats such as Flash-movies, images and PDF-files are exceptional. These are generally not adapted to the different presentational contexts in any way apart from being re-sized, if they're viewable at all. They lack the flexibility of HTML/XHTML-based web content, but gain the possibility of authentic design cybernetics through a WYSIWYG. For example in an image manipulation program the WYSIWYG-concept makes sense, as opposed to when editing text content of HTML/XHTML-pages.

Stand-alone WYSIWYG-editors are often used to produce web content, there the WYSIWYG-concept has the same downsides as in a CMS-context. WYSIWYG-usage when designing templates for whole web sites, by web professionals knowledgeable of document structuring, is on the other hand out of the scope of this thesis. The WYSIWYG-concept might be useless or even harmful for them too, despite modern WYSIWYG:s such as Dreamweaver being designed with web standards in mind. The findings here can't however be directly applied on that kind of usage, since they concern content editing and not template design. A skilled web professional also knows that the WYSIWYG-view is just a rough preview, and that browser testing is inevitable.

Structural WYSIWYG:s

The case study showed that even a WYSIWYG that emphasizes structural content tends to sometimes have its user marking up the content according to presentation instead of structure. This tendency to trick its user into marking up content according presentation (by viewing something close to it) could be considered a usability flaw, since such a WYSIWYG is indeed meant for structuring content. In the case study more thorough instructions to the content editor could have helped, but that would still not have been a remedy for the usability flaw itself.

One could of course try to prevent some of these mistakes from happening by having an even stricter WYSIWYG, in the case of the Bitflux Editor by configuring its schema to be even stricter about what markup is allowed in which contexts. This would however not address the actual usability flaw either, only in some cases prevent the user from succeeding to apply markup according to presentation.

Conclusion

WYSIWYG:s in general give the notion of control over presentation to its user and should not be used when this notion is false, which it usually is when editing web content. By providing tools for manipulation of structure while still showing presentation, a structural

WYSIWYG goes only half-way towards a solution. From a usability standpoint this kind behavior doesn't make any sense either.

The WYSIWYG-concept in itself seems to be the core problem, indirectly causing presentational markup and thereby inconsistently structured content. Therefore the widespread usage of WYSIWYG-editors in CMS-environments is highly questionable, and should be reconsidered. In fact since most kinds of text content today (including non-web content) should be re-usable in different presentational contexts, that content would be better off managed through a structural editor as well.

5.2.2 Alternatives to WYSIWYG

The usability flaw of structural WYSIWYG:s could be corrected by making a WYSIWYG show structure instead of how the page supposedly would look, but then it would have been made into what we know as a WYSIWYM. A WYSIWYM could also show elements more clearly and therefore avoid having the user mistakenly adding them, as opposed to a WYSIWYG which strives to show the eventual outcome, in the case of the Bitflux Editor leaving some of the changes made invisible to its user.

In some cases even editors more different from a WYSIWYG, such as one using wiki-style markup or something similar like Textile could be tried. Also some system built using traditional forms, having a different field for each element, could be tried out.

Using Textile instead of a WYSIWYG for the purpose of consistent markup seems to be a somewhat successful solution:

> I have set up a CMS for the intranet of a design-school here. I used Textile for html-formatting. About 20 professors and teachers with extremely different computer-skills are feeding content into this system. It turned out, that the code all those people produce in this CMS is highly consistent and mostly free of errors. None of these people had severe problems starting to use Textile, but of course they needed instructions.
>
> No WYSIWYG-editor works for people who don't know what they're doing. It would have taken the same or higher effort for those people to learn how to properly use a WYSIWYG-editor.
>
> [rai(2007)]

A WYSIWYM or traditional forms however would probably be more self-explanatory and require less or no instructions to the user.

In cases where the content editor should be allowed some control over the presentation one could look into the concept of Presentation Management Systems (PMS), rather than allowing the usage of less strict WYSIWYG:s. In most cases however the content editor only needs control over content structure, since design issues usually should belong to XHTML and CSS savvy web designers.

In any case there is always a possibility that the content editor might look at the visual outcome by checking a page in a browser right after editing it, therefore being able to do further adjustments according to the visual outcome. No technical restrictions such as having only a non-WYSIWYG structural editor provided could prevent this from happening. Only having a workflow involving a separate moderator for content approval could prevent browser previewing. Using a structural editor such as a WYSIWYM in case studies otherwise similar to the one discussed here could clarify whether this would pose a problem in reality. Monitoring of the user's behavior would make sense in such a test case, to verify whether test and editing iterations take place.

In addition to technical solutions more thorough education of the content editor than in the case study of this thesis could be wise, not only education about semantics but also about how to write good web texts in general. Education about semantics would be especially important if browser checking would cause problems concerning the proper usage structural editors.

Relying on a WYSIWYG-editor to solve these kinds of problems is not, then, a sensible solution. It will often create more, and more serious, problems, than it solves.

Bibliography

[WCA(1999)] Web content accessibility guidelines 1.0. www, May 1999. URL `http://www.w3.org/TR/WAI-WEBCONTENT/`.

[rai(2007)] Ruby on rails based cms in ruby on rails. www, August 2007. URL `http://wiki.rubyonrails.org/rails/pages/Ruby+on+Rails+based+CMS`.

[wik(2007)] Wysiwyg. www, August 2007. URL `http://en.wikipedia.org/wiki/Wysiwyg`.

[Berners-Lee(2000)] Tim Berners-Lee. *Weaving the Web: The Original Design and Ultimate Destiny of the World Wide Web*. Collins, 1 edition, 11 2000. ISBN 006251587X.

[Group(2004)] W3C Technical Architecture Group. Architecture of the word wide web, volume one. www, December 2004. URL `http://www.w3.org/TR/2004/REC-webarch-20041215/`.

[Ho(2005)] Tong Ho. A scriptable xml editor with multiple css configurable views. Technical report, The Faculty of the Department of Computer Science, San Jose State University, 2005.

[Initiative(2006)] Web Accessibility Initiative. Policies relating to web accessibility, August 2006. URL `http://www.w3.org/WAI/Policy/`.

[Lie and Bos(1999)] Håkon Wium Lie and Bert Bos. *Cascading Style Sheets: Designing for the Web (2nd Edition)*. Addison-Wesley Professional, 2 edition, 7 1999. ISBN 0201596253. URL `http://www.w3.org/Style/LieBos2e/history/`.

[Lombardi(2004)] Victor Lombardi. Integrating css with content management systems. www, September 2004. URL `http://www.digital-web.com/articles/integrating_css_with_cms/`.

[Meyer(2006)] Eric Meyer. *CSS: The Definitive Guide*. O'Reilly Media, Inc., 3 edition, 11 2006. ISBN 0596527330.

[Musciano and Kennedy(2006)] Chuck Musciano and Bill Kennedy. *HTML & XHTML: The Definitive Guide (6th Edition)*. O'Reilly Media, Inc., 6 edition, 10 2006. ISBN 0596527322.

[Nielsen(1997)] Jakob Nielsen. The difference between web design and gui design, May 1997. URL http://www.useit.com/alertbox/9705a.html.

[Schrage(2004)] Martijn Michiel Schrage. *Proxima: A presentation-oriented editor for structured documents*. PhD thesis, Universiteit Utrecht, October 2004.

[Spiesser and Kitchen(2004)] Jacqueline Spiesser and Les Kitchen. Optimization of html automatically generated by wysiwyg programs. Technical report, Department of Computer Science and Software Engineering, The University of Melbourne, 2004.

[Stocker(2005)] Christian Stocker. Bitflux editor users mailing list. www, November 2005. URL http://lists.bitflux.ch/pipermail/bx-editor-users/2005-November/000332.html.

[Taylor(1996)] Conrad Taylor. What has "wysiwyg" done to us? *The Seybold Report on Publishing Systems*, 26(2), September 1996. URL http://www.ideography.co.uk/library/seybold/WYSIWYG.html.

[Zeldman(2001)] Jeffrey Zeldman. *Taking Your Talent to the Web: Making the Transition from Graphic Design to Web Design*. Waite Group Press, 1st edition, 5 2001. ISBN 0735710732.

[Zeldman(2006)] Jeffrey Zeldman. *Designing with Web Standards (2nd Edition)*. Peachpit Press, 2 edition, 7 2006. ISBN 0321385551. URL http://www.zeldman.com/dwws/.