Bachelor Thesis
Computer Science

Radboud University

Snap-to-Style

or incorporating the benefits of structured documents
in WYSIWYG word processors

Author
Daniel Roeven
s4356772

Supervisor & first assessor
prof. dr. Erik Barendsen
e.barendsen@cs.ru.nl

Second assessor
dr. Sjaak Smetsers
s.smetsers@science.ru.nl

January 25, 2016

Abstract

Structured documents and their editing environments offer advantages that WYSIWYG word processors might benefit from. We identify two ways of offering these advantages to writers using WYSIWYG systems. We do this by observing how writers currently work in WYSIWYG systems and by creating and evaluating a prototype implementing some of these advantages. The prototype offers the advantages in two manners: the use of preset styles is encouraged and the direct formatting controls are repurposed to display the functionality of preset styles. Results indicate that encouraging the use of preset styles leads to improved usability and better document formatting. Repurposing direct formatting controls is promising, but further refinement of the mechanism and evaluation is necessary to confirm this.

Acknowledgements

I express my sincere gratitude to my supervisor Erik Barendsen for his guidance. For the pick-me-ups, encouraging or helping me wherever I felt inexperienced and unsure. And for pointing me towards the right track, refining my ambitions whenever I was a tad too enthusiastic.

I would also like to take this moment to thank my family. My mother, father and little brother for their unfaltering support and confidence — not only during the writing of this thesis, but during my entire education. I might add that it extends even beyond that.

A thank-you to my close friends for putting up with my woes and lending me their sharp eyes when proofreading. My fellow informaticians and the odd anthropologist: Jaco, Marte, Sander, and Timo — here's to you.

Contents

Introduction

1

The two dominant approaches to word processing are WYSIWYG editing and visible markup (Chamberlin, Hasselmeier, & Paris, 1988).

Microsoft Word is a widely-known and widely-used word processor. It adheres to the WYSIWYG approach: What You See Is What You Get. The representation of the document being created is visually equal to the end product; writing in such an environment means creating content and presentation at the same time. The writer has complete freedom of choice in layout and formatting. The word processor is thus aware of all presentational aspects, but not necessarily all structural aspects of the document.

LaTeX is a well-known document preparation system in academia, primarily in the stem fields (What are TeX and its friends?, 2016). It is a structured document editor, meaning that one explicitly does not focus on the presentational aspects, but only on the content. The content and its representation are one step removed from each other. The writer indicates structure by marking up the document according to computer-readable codes, appropriately called markup. The presentational aspects (layout and formatting) are handled by the environment. LaTeX documents are written according to tagging conventions of the LaTeX markup language to demarcate structure. Its typesetting algorithms then provide formatting and layout.

Both approaches have their merits. WYSIWYG approaches require very little technical expertise and enable the writer to create documents without delving into the specifics of the file format. Structured documents are marked up according to computer-readable codes, enabling the editing system to automatically format the document in a consistent style. This does, however, require a certain technical expertise and proficiency from the writer.

1.1 AIM OF THE STUDY

Introducing structure to your document via descriptive markup has a number of benefits (Johnson & Beach, 1988; Flynn, 2014, p. 10).
First and foremost, formatting, layout, and style are consistent throughout the entire document. The computer-generated formatting displays no consistency errors.

Secondly, styles allow the writer to change multiple properties of a document element with a single action, rather than changing each property separately. For example, when changing a heading to another heading, rather than changing font family, size, and weight, only the heading type needs be changed. The editing system will then take care of the rest.

Similarly, an entirely different style can be applied, matching the same labels to a new format or layout. It is a low-effort endeavour to switch to a different layout, should the writer wish to do so. It is common, for example, when submitting to academic journals with given formatting guides.

Finally, any computer-based searching or browsing can return a valid reference to the place located within the document. This is a great benefit when cross-referencing, creating bibliographies and appendices, adding footnotes, and most commonly, adding a table of contents.

Despite the advantages, use of markup and structured documents is not widespread and remains primarily known in the stem fields. These advantages would benefit all writers, not just those in academia. This thesis aims to identify an editing approach offering these benefits to writers working in WYSIWYG word processors. Without requiring the user to learn to use an entirely unfamiliar system, we explore whether an editing approach can be created in which the system is aware of the underlying document structure. Such a system should accessible to the technically-gifted and technical laymen alike, while reaping the benefits of structured document approaches. Concretely: *how can a WYSIWYG word processor be designed to incorporate the benefits of structured documents?*

To answer this we will first observe and interview participants working in a WYSIWYG word processor to gain insights into how writers currently work in such environments. We answer the subquestion: *how do writers currently work with WYSIWYG word processors?*

Given these insights, we will leverage knowledge from the structured authoring domain to improve WYSIWYG environments. Cherry-picking the benefits from structured documents and applying these to WYSIWYG environments. Once such an approach has been identified, a small prototype will be constructed and tested, as proof of concept, answering the subquestion: *does combining the knowledge of structured documents and the insights into WYSIWYG editing lead to improved usability and document formatting?* Should the approach prove fruitful, it can be applied to larger-scale word processors.

1.2 BACKGROUND

We examine existing literature on markup, structured documents, and WYSIWYG editing.

But first, a small matter of linguistics: a clarification of the terms used to indicate the different parties involved in this thesis. The obvious term to use for someone who uses a computer system is a “user”. Usability experts, however, have long been calling for another way to refer to such a user, lest

we forget that these users are humans, and not machines. However, a one-for-one replacement of "user" with "human" makes for awkward sentences. We therefore refer to users of word processors as *writers*. When referring to the writers that participated in tests and interviews for this thesis, we use the term *participants*. And finally, when referring to ourselves, we say *authors*.

1.2.1 *The many taxonomies of markup*

As mentioned previously, the two major approaches to document formatting are *direct manipulation* (known today as WYSIWYG systems) and *visible markup* (Chamberlin et al., 1988).
Flynn (2014, p. xxv) defines markup as:

> "The tags, entity references, control sequences, escapes, commands, and other special values in a document which are not part of the text but which serve to identify the component parts of it, or specify what to do with it."

This definition rings familiar to our intuitive understanding of markup. But it is possible to identify multiple different species of markup. A simple yet succinct division is the visual/logical divide by Lamport (1986), based on earlier work by Reid (1980) and Roberts (1980):

Visual markup solely describes the visual aspects of the document elements it demarcates. Its tags only contain formatting or layout information. No semantic information is captured. The HTML <i> tag (for italics) is an example of visual markup.

Logical markup describes the semantic aspects of the document elements it demarcates. Its tags describe a document element as Title, Heading, List, etc. The HTML tag (for emphasis) is an example of logical markup (often rendered in an italic font style).

For the purposes of this thesis, however, we need another distinction. Indeed, there are many more categories that we can readily use. Coombs, Renear, and DeRose (1987) speak of many different categories of markup (even considering interpunction and capitalisation a form of markup). Of main interest here, however, are the three categories they call presentational, procedural, and descriptive.

Presentational markup encompasses the actions a writer uses to make the presentation clearer. Such markup includes spacing, page breaks, page numbers, etc.

Procedural markup encompasses a set of commands written interspersed in the document, such as <i>, indicating how text should be formatted (italic, in the case of <i>).

Descriptive markup is, similarly, a set of commands written interspersed in the document. In contrast with procedural markup, however, it does not indicate how text should be formatted, but rather identifies different element types like heading, paragraph, and list. Such markup is then evaluated according to a set of style rules to determine format and layout.

Logical and descriptive markup are roughly equivalent: they provide a description of the content and its structure, but make no claim on how it should be formatted. Presentational and procedural markup both describe visual aspects of the document and are therefore a subset of visual markup. The distinction between presentational and procedural markup, however, will become relevant shortly. See figure 1.1 for a diagrammatic overview of the different markup categories.

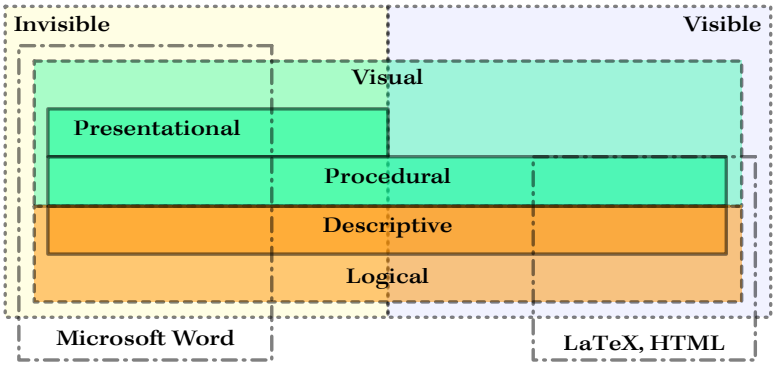

Figure 1.1: a classification of different markup categories and associated editors or file formats.

Markup in structured document environments We briefly discuss the role of markup in two major structured document standards, namely LaTeX and HTML. While LaTeX does include some presentational markup (automatic page numbering, standard margins), if one wishes to change such presentational aspects, explicit markup must be placed in the document. HTML does not include any such presentational markup from the get-go.

Both procedural and descriptive markup are present in both LaTeX and HTML. As an example of procedural markup, `\textit{foo}` and `<i>foo</i>` are used in LaTeX and HTML respectively to set the word "foo" in an italic font style. But an equivalent in descriptive markup exists too: `\emph{foo}` and `<em>foo</em>`. These are the tags to place emphasis on the word "foo" in respectively LaTeX and HTML. Visually similar (emphasis is often set in an italic font style), but categorically different.

Markup in WYSIWYG environments It might sound like an oxymoron to speak of markup in WYSIWYG environments. While indeed the writer sees

no markup tags (and perhaps is not even aware of them), the underlying file format certainly adheres to some markup standards. This is maintained by the WYSIWYG environment itself.

Documents created in a WYSIWYG word processor are at the very least marked up presentationally: the writer indicates how the document should look. The writer does not see any tags or control sequences in the document, but they are most certainly there. They are discretely managed by the word processor, invisible to the writer.

This does not mean that documents created in a WYSIWYG word processor are only marked up presentationally. They can be marked up procedurally, too. For example, when a writer uses a bold font, the selected text is shown as bold to the writer. Once again, invisibly, the word processor adds a bold tag to the underlying file.

Finally, even descriptive/logical markup can be achieved in a WYSIWYG word processor. This is done via the use of preset styles (more on this in section 1.2.2). When a writer applies a Heading 1 style to some element in the document, an appropriate tag is added to the underlying file. This is easily evidenced by formatting a document in Microsoft Word only via preset styles, and exporting it to the HTML file format. Headings are then indicated with an appropriate h1 tag, paragraphs with the p tag, and so on.

1.2.2 *WYSIWYG editing*

There are different ways of formatting in WYSIWYG environments. Johnson and Beach (1988) identify two principal approaches, namely via the direct manipulation of individual typographic properties, or via the use of preset styles.

Direct formatting For practicality's sake we shall refer to the lengthy "direct manipulation of individual typographic properties" as *direct formatting* (not to be confused with the term "direct manipulation" coined by Shneiderman (1983), which refers to any WYSIWYG editing and is now a well-known term in the field of Human-Computer Interaction). When formatting a heading directly, one individually sets the typographic properties as font family ("Helvetica" or "Times New Roman"), font size (12 pt. or 24 pt.), font weight (bold or regular), font style (italic or roman), etc. As we have seen in 1.2.1, this largely corresponds to procedural markup, even though the writer might be unaware of such markup.

Preset styles We shall refer to the latter of the two formatting approaches as formatting by use of preset styles. Johnson and Beach (1988) and the Microsoft Word documentation (Style basics in Word, n.d.) simply call this

"styles". Sørgaard and Sandahl (1997) call it "paragraph styles". We deviate from this nomenclature. Paragraph styles implies that it only applies to paragraph text. Simply "styles" is ambiguous; we often speak of font style, by which we indicate whether text is italic or not.

Preset styles accurately reflect their role in WYSIWYG processors such as Microsoft Word. After all, by the press of a single button, a preset style of typographical properties is applied to the selected element. Once again referring to 1.2.1, this approach corresponds to the use of descriptive/logical markup. However, as Sørgaard and Sandahl (1997) have found, its use is not widespread among WYSIWYG writers.

Methodology

2

2.1 STRUCTURE OF THE STUDY

The structure of this thesis is based on the User Centered Design (UCD) process, as documented in ISO standard 9241 (ISO 9241-210, 2010). This standard outlines the six key principles:

1 The design is based upon an explicit understanding of users, tasks and, environments.
2 Users are involved throughout design and development.
3 The design is driven and refined by user-centered evaluation.
4 The process is iterative.
5 The design addresses the whole user experience.
6 The design team includes multidisciplinary skills and perspectives.

This research consists of four stages. The first stage takes a look at the *status quo* of document editing. It will consist of a series of interviews and participant observation tests with writers on their use of WYSIWYG word processors. This stage and the background literature corresponds to the first principle.

The next three stages will consist of design-and-creation cycles, in which we attempt to apply the knowledge of writers and processors gained in stage one, in order to outline the design of a change or addition to existing writing environments. This will amount to a small prototype of a limited subset of word processing features. Each stage calls for feedback (on usability and features) from actual writers. Such feedback will be provided by the same writers as in stage one via small usability tests. These stages address principles two, three, and four.

Principle five and six are difficult to realise fully in a Bachelors thesis. Due to time constraints and limitations of scope only a small prototype focused on refining a small aspect can be made. Also, the design team consists of only one person from one discipline.

However, the improved aspect is part of a larger context (WYSIWYG word processors), which indeed does address the whole user experience. We simply do not need to address the whole user experience.

Secondly, the field of research with which this thesis is concerned is the field of Human-Computer Interaction (HCI). This field focuses on the design and use of computer technology by humans. It exists at the intersection of many disciplines, amongst which psychology, industrial design, and chiefly, computer science. By its very nature this research is multidisciplinary.

Given the structure of this research, it is not possible to accurately describe the method of analysis for each stage at this point. The design and evaluation of each stage is based upon the results of the previous stage. We therefore describe the method and results for each stage separately. This amounts to four distinct methods and results for each of the four stages. A general overview is given here.

The first stage consists of finding out how writers currently work in WYSIWYG environments. We do this by a form of participant observation to get an overview of which affordances are used, what their workflow consists of, and what their grievances are while working in a WYSIWYG processor. Five writers are asked to reproduce a document in Microsoft Word. They are observed during the writing in a moderated task test, a post-test interview is conducted with the writers, and finally the produced document is analysed for formatting and layout.

The second stage consists of combining the knowledge of structured documents and insights into WYSIWYG editing into a best-of-both-worlds editing approach. We develop a prototype and evaluate it on usability and verify that it indeed offers the benefits of structured documents. This is done via a moderated task test, a post-test questionnaire, and an analysis of the produced document.

The third stage consists of refining the prototype based on the feedback from the second stage. It is evaluated in the same manner as stage II.

In the fourth and final stage, we specifically test one aspect of the refined prototype. It is evaluated in the same manner as stage II and III.

Stage I: Exploratory study

3

The first step in identifying ways of applying the benefits of structured documents is to get insights into how writers work in WYSIWYG software. This corresponds to the first principle of user-centered design: *the design is based upon an explicit understanding of users, tasks, and environments*. We begin by examining writers working in their WYSIWYG editor of choice, answering the subquestion: *how do writers currently work with WYSIWYG word processors?*

3.1 METHODOLOGY

Two techniques were used: job shadowing and contextual interviews. Job shadowing is a form of participant observation. It consists of unobtrusively observing users while they work in the manner they normally would (Mathis, 2011, p. 10). Contextual interviews are post-test interviews about the how and why of their work (Mathis, 2011, p. 11).

Five participants are enough to find the majority of all usability problems (Nielsen, 2000). Five participants of different educational backgrounds were chosen. All participants are acquaintances of the author. All participants were enrolled in higher education at the time the tests were conducted. Their fields of study are Biology, Industrial Design, Journalism, Law, and Teacher Education. They were all of ages 18 through 22 at the time of participation.

Before the start of the test and interview, the participants were verbally notified of the purpose of the research (to improve WYSIWYG word processors), the role they play in it (write in word processors and evaluate them), what happens to their personal data (is viewed by the author and assessors but kept private), and what the information gathered during the tests is used for (analysis by the author and assessors). They were informed of the possibility to opt-out of the research at any time. The guideline used to verbally notify the participants is to be found in the appendix section A1.

3.1.1 *Job shadowing*

The participants were asked to create a document in a WYSIWYG editor, in the same manner as they do when writing a deliverable assignment for their education. As the job shadowing method prescribes, the observer sits behind them, taking note of their actions.

To introduce consistency amongst the different participants, we have created a template document which they reproduce. This template displays many different document elements available in word processors. These elements are then reconstructed by the participants, giving insightful

information as to how these reconstructions are achieved in the editor. The template was handwritten in order to reduce emphasis on visual presentation; it is impossible to precisely graphically duplicate a handwritten document in a digital environment. The template is available in the appendix section A2. The template contains the following document elements:

- Title
- Author
- Date
- Headings one, two, and three
- Paragraphs
- Bold text
- Italic text
- Quote
- Lists
  - Ordered
  - Unordered
  - Multiple depths
- Bibliographical citation

An observation sheet was used by the observer to denote the actions performed, ordered by document element. See appendix section A14 for the observation sheet.

3.1.2 *Contextual interviews*

Upon completion of the test, a small interview was conducted. This interview's purpose was threefold: to confirm observations garnered during the job shadowing; to gauge completeness of the job shadowing (are there things you often do that we did not see here?); and to map their preferred and disliked interaction with the word processor.

The interview explicitly did not focus on what writers want, would like, or need. As experts agree, users have notoriously poor judgment for these kind of things (Gócza, 2010). Psychologists call this phenomenon the introspection illusion (Pronin, 2009).

The interview was set up in a semi-structured fashion. Questions have been prepared in advance to provide a guideline for the interview (see appendix A8), but there was also room for relevant additional questions that surfaced. The interview is shaped as follows: it begins with very general questions (allowing for the branching off of different subquestions, while not leading the interviewee towards "preferred" answers) and steadily progressing towards specific questions pertaining to certain usage scenarios.

3.1.3 *Produced documents*

The participants were asked to share the created document with the author via email. This allows for an analysis of the document features and structure.

3.1.4 *Data analysis*

The three different sources of data (documents, observations, interviews) were coded separately using open coding. The notes made on observation sheets (appendix A14) during the test were coded inductively.

The three purposes of the interviews were confirming observations of job shadowing, gauging completeness, and discovering their problems and frustrations. The interviews were transcribed and coded both inductively, to gauge completeness and discover problems/frustrations, and deductively, to support the notes from the observations.

The documents created during the test have been annotated with whitespace characters (tabs and returns), document elements (hierarchy and pool), typographical information (font family, size, weight and style), and assigned preset style in Microsoft Word. These items come from the background literature of WYSIWYG and structured authoring environments. Whitespace formating falls under the category of presentational markup. Hierarchy elements are elements that convey structural information to the reader, such as the title and headings. Pool elements are the actual content, such as paragraphs, lists and quotes. These terms are borrowed from Flynn (2014, p. 18). Assigned preset style in Word is an instance of invisible descriptive markup. Finally, if the typographical information changes among different document elements while the preset style does not, it signifies the use of presentational markup (or direct formatting).

3.2 RESULTS

The coding of the three separate data sources are presented in three separate tables in the appendix. Below follows an explanation of which codes were used and how they are categorised and presented in the tables.

3.2.1 *Job shadowing*

The results from the coding of the job shadowing observations are presented in table *Observation analysis stage I*, to be found in appendix A20. Columns contain the codes used when categorising the observation notes. These categories are:

- *Typography*
  - *Bold/Italic*
  - *Alignment*
  - *Font size*
- *Headings*
- *Returns*
- *Lists*

Rows contain participants: *Adam, Eva, Louisa, Robert,* and *Timothy*. The participants's names have been anonimized. Each entry in the table contains a reference to the corresponding table cell in the relevant observation sheet.

3.2.2 *Contextual interviews*

The transcriptions can be found in the appendix A9-A13. The results from the coding are presented in table *Interview analysis stage I*, to be found in the appendix A21. Each entry in the table contains a reference to the relevant interview line number. Again, rows contain participants and columns contain categories of codes used:

- *Completeness*
  - *Does not normally do but occurred in test*
  - *Does normally do but did not occur in test*
- *Document elements*
  - *Headings*
  - *Returns*
  - *Lists*
- *Editors*
- *Usage*
- *Problems/Frustrations*
- *Looks*
- *Separation of presentation and content*

The codes *Completeness, Editors, Usage, Looks,* and *Separation of presentation and content* pertain to the background of the participants and completeness of the test, *Document elements* and *Usage* support the notes made during the observation, and *Problems/frustrations* to any disliked interaction with the editor.

3.2.3 *Produced documents*

The annotated documents can be found in the appendix.

- Concerning whitespace characters: tabs are indicated with a purple rightwards arrow (→), newlines with a purple pilcrow sign (¶), and soft newlines with a purple leftwards arrow with hook (↩).
- Concerning document elements: hierarchy elements are indicated with a green underline, pool elements are indicated either by yellow underline (quote) or by yellow outline (paragraph, list).
- Concerning typography: on the right hand side of each green or yel-low line the typographical information has been displayed in the form document element name, font family, font size, font weight, font style.
- Concerning assigned preset style: on the left hand side the style as demarcated in Microsoft Word is displayed.

This information is summarized in table *Document structure analysis stage I*, available in appendix A22. Rows refer to each participant's documents. The different document annotations are put into columns:

- *Styling*
- *Alignment*
- *Returns*
- *Consistency*

3.2.4 *Findings*

The three tables as described above now display information on usage of and problems with WYSIWYG word processors. Findings that emerge from frequent entries in the tables are described, grouped by subject. These groups are:

- **Test completeness**
- **Background experience and processor choice**
- **Direct formatting versus preset styles**
- **Order of work**
- **Document look**
- **Whitespace**
- **Ordered and unordered lists**
- **Consistency**
- **Problems and frustrations**

The findings are described with references to the table in which they appear. References are denoted as *Table name, Column name, (optionally) Participant*

name. An occasional image or quote is shown for emphasis and clarity. Quotes are translated from Dutch by the author. Translation and original are shown side-by-side.

Test completeness While some participants indicated they performed actions in the test they do not do normally, and conversely, that they normally perform actions which did not occur in the test (interview analysis, column completeness), these occurrences were rare and did not form any pattern. Save for these outliers, the test is adequately representative of their normal usage of word processors.

Background experience and processor choice All five participants used Microsoft Word (on either Mac or Windows) to create the test document (interview analysis, column editors). None of the participants are actively looking for other editors (interview analysis, column editors), despite their frustrations. They all use Microsoft Word mostly for education-related purposes (interview analysis, column usage). Despite their different backgrounds and educations, usage of Microsoft Word was very similar and adequately captured in this test.

Direct formatting versus preset styles Four of the five participants formatted their documents directly. This is evidenced by the document structure analysis, columns styling and alignment, observation analysis columns typography and headings, and interview analysis column headings. They did thus not introduce descriptive markup to their document. Participant Louisa used preset styles.

Eva formats directly and writes the table of contents by hand (interview analysis, column headings, participant Eva). Conversely, Louisa, who formats everything with appropriate preset styles, often automatically generates the table of contents (interview analysis, column headings, participant Louisa). Among those who directly formatted, two approaches were used. They either recreated each style by hand, or they reused styles by copy-and-pasting from previous styles (observation analysis, column headings).

Reasons for not using preset styles include that it does not look good (interview analysis, column looks, participants Robert and Timothy) or not knowing how they work (interview analysis, column usage, participant Eva).

Order of work All five participants wrote the entirety of the text before formatting and styling their document (interview analysis, column usage).

Document look Four of five participants (everybody but Louisa) place importance on document look and all participants are content with how

their produced documents look (interview analysis, column looks). Two participants achieve good-looking documents by purposefully refraining from using preset styles and directly formatting their documents (interview analysis, column looks, participant Robert, Timothy):

> The new version of Word has styles, and they might work, but nah, they're not very pretty.
>
> Die nieuwe versie van Word, daar staan stijlen, misschien werkt dat wel, maar nah, dat is niet mooi.
>
> Interview with Timothy

Whitespace Four of five participants (everybody but Louisa) use newlines as whitespace, rather than purely as content separator (document structure analysis, column returns, observation analysis column returns). These are the four participants that directly format their document. The most common use of a newline as whitespace was to introduce a bottom margin after a hierarchy element.

The one participant (Louisa) who makes use of preset styles uses newlines only as content separator (document structure analysis, column returns participant Louisa). She says to try and use as few newlines as possible (interview analysis, column returns, participant Louisa):

> [. . .] in the menu, insert and then page break. I never add enters, because that doesn't work. [. . .] I try to do that as little as possible, adding enters.
>
> [. . .] in het menu en dan invoegen en dan page break. Ik doe nooit enteren, want dat werkt niet. [. . .] Dat probeer ik zo min mogelijk te doen, enteren.
>
> Interview with Louisa

Instead of using newlines as means of inserting a new page, she uses page breaks. Especially relevant (by means of stark contrast) here is participant Eva, who uses four consecutive newlines to outline document elements to the bottom of the page (see Eva's document). See figure 3.1 for clarification.

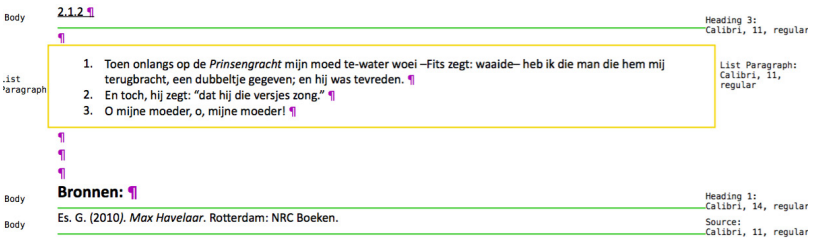

Figure 3.1: using newlines as whitespace. From Eva's document.

Adam uses tabs as means of center-aligning or indenting text (see Adam's document or figure 3.2 for clarification). Tabs as means of center-alignment can lead to layout errors also. Should the length of the title change (by changing the amount of characters or by use of a different font family or size), it would no longer be center-aligned.

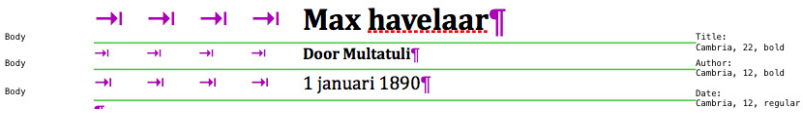

Figure 3.2: using tabs for center-alignment. From Adam's document.

Ordered and unordered lists Lists were often set in an appropriate preset style. In three cases this was due to automatic list recognition, a Microsoft Word feature in which lists are automatically converted to preset list style. In two cases a menubar option for lists was used to demarcate the preset list style (observation analysis, column lists).
In two tests the automatic list recognition did always not work, in which case the appropriate list style was not used (observation analysis, column lists).

Consistency Typographic inconsistencies of varying degrees occur in test documents. In Adam's and Timothy's case two headings of different levels had no typographical distinction. Adam also had one heading in two different typographical styles (document structure analysis, column consistency, and Adam's and Timothy's documents).

In four cases indent levels in different lists did not match each other (document structure analysis, column consistency).

Luisa, who made use of preset styles, produced a document free of consistency errors (document structure analysis, column consistency).

Problems and frustrations Three participants reported trouble with finding features (interview analysis column, problems/frustrations).

Two participants indicated they do not understand features (interview analysis, column problems/frustrations, participant Adam; interview analysis, column usage, participant Eva).
Two participants experience frustration when the layout suddenly changes (interview analysis, column problems/frustrations):

Well, it's just a little odd, sometimes things move about and you can't get them on the line you want them to, no matter what you do.	Ja het is gewoon een beetje vreemd, soms verspringen er allemaal dingen die je niet wil laten verspringen en die je op geen enkele mogelijke manier weer op die regel kunt krijgen zoals je het wil hebben.

Interview with Robert

3.2.5 *In summary*

The test reflected their normal usage of word processors.

Four participants format directly and use newlines as whitespace. The documents created by these participants display typographical consistency errors. One participant uses preset styles and does not use newlines as whitespace. Her document is free of typographical errors. All participants write content first and format afterwards.

Lists were often properly demarcated, even by those who directly formatted, due to automatic style recognition.

Four of five participants place importance on document look. All participants are satisfied with how their documents look. Two participants achieve good-looking documents by purposefully not using preset styles. Usability problems exist in finding and understanding features. Participants also experience frustration when layout suddenly changes.

Stage II: Prototype design and evaluation

4

We now have some concrete knowledge of the goings-on in WYSIWYG writing. Does combining the knowledge of structured documents and the insights into WYSIWYG editing lead to improved usability and document formatting?

We have seen that preset styles and descriptive markup are two manifestations of the same strategy in different editing approaches. They provide a way for the editing environment to keep track of the document's structure.

However, preset styles are not used by many writers. In the previous test only one out of the five participants used preset styles. Four participants formatted directly. A low adoption rate is consistent with Sørgaard and Sandahl's (1997) findings.

We argue that using preset styles will remedy the problems we have found in the previous stage. First, we briefly restate the benefits of preset styles (invisible descriptive markup) here. Then we show their relevance to the problems encountered in the previous stage. The benefits of preset styles include (Flynn, 2014; Johnson & Beach, 1988):

1 Formatting, layout, and style are consistent throughout the entire document.
2 Styles allow the writer to change multiple properties of a document element with a single action, rather than changing each property separately.
3 An entirely different style can be applied, matching the same labels to a new format or layout.
4 Any computer-based searching or browsing can return a valid reference to the place located within the document.

As the the findings show (3.2.4 **Consistency**), documents are not free of consistency errors. Formatting with preset styles reduces these consistency errors.

Four participants format with newlines (**Whitespace**). Some frustrations of layout changing unexpectedly (**Problems and frustrations**) are to blame on newline-formatting. Using newlines as whitespace can lead to layout errors. See figure 3.1. If, for example, Eva would have added text to the paragraphs above, this would have pushed the bottom-outlined text onto a new page, resulting in two half-filled pages rather than one fully-filled page. With properly designed preset styles, formatting with newlines is necessary no longer. Appropriate whitespace is applied as bottom margin of a heading when using preset styles. Rather than manually inserting newlines after each heading, the preset styles take care of it.

Another ability enabled by the use of preset styles is the ability to automatically construct a table of contents. This is indeed reflected by the

finding that Louisa, making use of preset styles, does automatically generate tables of contents, while Eva, who directly formats, does not (see **Direct formatting versus preset styles**).

4.1 SNAP-TO-STYLE

Rather than forcing the writers to use preset styles, we apply the benefits of preset styles to direct formatting. That way writers do not need to change their workflows but do receive the advantages.

We propose a system in which writers can format directly, but descriptive markup is *inferred* from their actions. This is done by disallowing a continuous field of options concerning typographic properties. Rather, it is limited to a discrete and predetermined set of styles. When changing one property, other properties are automatically adjusted too, so that only preset styles can be achieved by direct formatting. Informally, when a writer increases the font size, it "snaps" to the next enabled font size (corresponding to a preset style), rather than enumerating each size in between. When a writer enables bold type on a heading, it snaps to the nearest heading that has the bold type property. Of course, formatting with preset styles should still be possible in this system.

For the inferring of descriptive markup from direct formatting, we identify the following direct formatting actions along with their corresponding affordances (ways of carryings out said action):

- Setting a specific font size. Affordance: dropdown menu.
- Increasing font size. Affordances: button + or keyboard shortcut `ctrl +`.
- Decreasing font size. Affordances: button - or keyboard shortcut `ctrl -`.
- Toggling weight (turning bold type on/off). Affordances: button B or keyboard shortcut `ctrl b`.
- Toggling style (turning italic type on/off). Affordances: button I or keyboard shortcut `ctrl i`.

For each document element, there are three properties which can be directly formatted: weight, style, and size. Weight can be adjusted in precisely one way. The element either is in bold type, or it is not. Pressing the B button (or using the keyboard shortcut) will toggle the weight, enabling or disabling bold. Same for style: italic is a toggle-only property. This means that for any given element, there is one element to be reached by toggling the weight, and one element to be reached by toggling the style (visualised in 4.1 as an outwards arrow). Size can be adjusted in two ways: it can be enlarged or reduced. That means that for any given element there are two elements to be reached: one by enlarging the size and one by reducing the size. Save, of course, for either end of the scale. Enlarging the title's font size or reducing

the paragraph's font size is not possible (visualised by a self-referencing arrow). See 5.2 for a diagrammatic overview.

Concerning a font size dropdown menu, rather than displaying all sizes, it only displays the sizes corresponding to a preset style. When applying a font size from the dropdown menu, then, would apply the corresponding style.

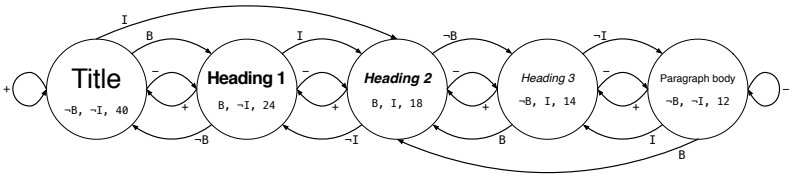

Figure 4.1: Diagram of direct formatting actions and their corresponding preset style.

Finally, a word processor implementing such a system should not remove the possibility of applying preset styles via dedicated buttons entirely. A hallmark of usable interfaces is flexibility: being able to perform the same actions via multiple ways (Mathis, 2011, p. 77). In keeping with the philosophy of not forcibly changing user behaviour, we need to support these affordances. Users such as Louisa, who format using preset styles in Microsoft word, should be able to use the same formatting workflow. Thus, five buttons for the preset styles will be included.

4.1.1 *Implementation*

A prototype implementing the described behaviour was developed in the web-based WYSIWYG editor Summernote (Summernote - Super Simple WYSIWYG Editor..., 2015). See figure 4.2 for a screenshot of the prototype. The buttons in the top left corner are the direct formatting controls, mapped to preset styles according to the Snap-to-Style approach. The buttons in the top right corner apply preset styles immediately. The button at the bottom completes the editing and generates the HTML code.

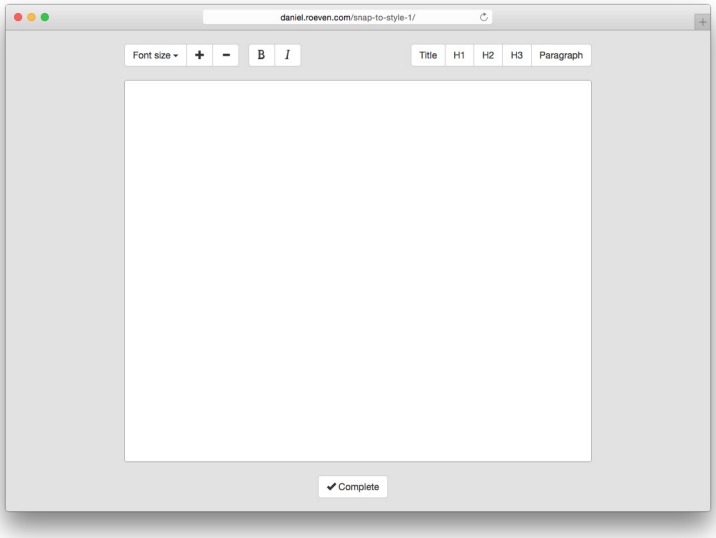

Figure 4.2: Screenshot of Snap-to-Style prototype. Available at daniel.roeven.com/snap-to-style-1.

4.2 METHODOLOGY

The prototype was evaluated on two criteria: usability and quality of produced document formatting. Usability was tested via a moderated task test and a post-task questionnaire. Quality of document formatting was done by analysis of produced documents.

Three participants have tested the prototype. After the first three tests, enough improvements could be identified to be implemented in the subsequent iteration.

4.2.1 *Moderated task test*

A moderated task test was conducted in which we asked the participants to reproduce a handwritten document of similar length to the previous document. The document is included in the appendix section B1. This is very similar to the job shadowing in the previous test, with the difference being that its focus now was testing the new interface and underlying formatting model, rather than mimicking the participants regular usage of word processors. Another important difference is that the prototype does not contain all the features that Microsoft Word contains. We therefore do not test elements: author, date, quotes, ordered lists, unordered lists, bold in running paragraph text, italic in running paragraph text. Observations on usage are made by the moderator during the test.

4.2.2 *Post-task questionnaire*

After the moderated task test, a small post-task questionnaire was conducted. The questionnaire consists of the ten questions from the System Usability Scale (SUS), devised by Brooke (1996). Then, two questions pertaining to the proposed system specifically are posed. These questions compare the proposed system to their regular WYSIWYG editor, which changes they have noticed and whether they appreciate these changes. Questions are included in the appendix section B6.

4.2.3 *Produced documents*

The written documents are saved as HTML, as this is Summernote's underlying file structure. As we have seen in section 1.2.1, HTML is capable of both procedural and descriptive markup. The resulting documents will display descriptive markup. This, however, is not the assurance of quality; we design our prototype such that procedural markup is not attainable. An assurance of document formatting quality is therefore the absence of typographical consistency errors and whitespace formatting, which are the issues found in usage of existing WYSIWYG editors (see section 3.2.5) and which are exactly the issues the proposal addresses.

4.2.4 *Data analysis*

The three different sources of data (documents, observations, questionnaire) were analysed as follows. The observations were coded deductively. The questionnaire was summarized in a table, comments for particular questions included, and a normalized SUS score was calculated. The produced documents were analysed for errors.

4.3 RESULTS

The coding or analysis of each of the three sources of data is presented in a table, included in the appendix. Below follows an explanation of which codes were used and how they are categorised or how the analysis took place. We then combine these three sources of data and present our findings grouped by subject, with references to the tables whence the findings came.

4.3.1 *Moderated task test*

The results from the coding are presented in table *Observation analysis stage II*, to be found in appendix B5. Columns contain the codes used when

categorising the observation notes. These categories are:

- *Direct formatting*
- *Preset styles*
- *Newlines*
- *Formatting/consistency errors*
- *Usage*

These categories represent the different usage options in the prototype, and the improvements that the prototype should enable. Also included is a general category *Usage* for any additional usage information. Rows contain participants: Eva, Robert, and Timothy.

4.3.2 *Post-task questionnaire*

Writers's answers and remarks to the SUS questions, as well as the two additional questions, are displayed in *Questionnaire Stage II*. The numerical answers are highlighted in green if the answer is positive, yellow if neutral, and red if negative. A normalized SUS score was calculated, according to the instructions from Brooke (1996). The score runs from 0 through 100, with 0 meaning very poorly usable and 100 very usable.

4.3.3 *Produced documents*

The produced documents have been annotated similarly to the annotations made in stage I, but altered to suit the new medium (HTML rather than Word documents). The annotations made are markup error, formatting errors, editor errors and interface errors. The first two are a result of writer behaviour, the latter two are a result of the software implementation. Below follows an explanation of each annotation. The annotations are presented in table *Document analysis stage II*, sorted vertically by participant, horizontally by category of error:

- *Markup error*
- *Formatting error*
- *Editor error*
- *Interface error*

Markup errors are errors where the implied heading (for example, the title at the top of the page) was not marked up as such in the HTML. Errors of type are grave: they indicate a mismatch between the user's perceived style and structure and the file's representation of said structure.

Formatting errors are where newlines are used as whitespace. As we have argued, formatting with whitespace is unnecessary and sometimes

problematic, causing layout to jump around unexpectedly. The prototype aims to reduce whitespace formatting by providing styles with appropriate margins above and below. Errors of this type are less grave than the previous, but nevertheless something to be reduced as much as possible.

Editor errors are less grave still. Whenever users type a space, followed by a newline, the editor stores this as a non-breaking space. Non-breaking spaces are a symptom of WYSIWYG file structure approaches: they contain information about the formatting. They are unnecessary: the content (HTML) is presented using a ruleset (css), which ensures proper formatting. The editor should simply not store this non-breaking space, in keeping with the separation of presentation and content, as well as general cleanliness.

Interface errors are usability errors. Writers often typed many newlines at the end of the document — or rather, at the end of the viewport of the editor window. When typing text, the viewport automatically scrolls down. When inserting newlines, however, it did not automatically scroll down. This resulted in many redundant newlines, which we do not count as formatting errors, but as interface errors.

The documents created during the test have been rendered according to the same ruleset as in the interface. This ruleset is not included in the created HTML document. Thus, the rule <link href="style.css" rel="stylesheet"> was added manually for analysis, at the top of the document. For clarity, it was marked with an orange background.

4.3.4 *Findings*

The three tables now display information on usage of and problems with the prototype. Findings that emerge from frequent entries in the tables are then described in groups:

- **Consistency**
- **Whitespace**
- **Implementation errors**
- **Direct formatting controls versus preset style buttons**
- **Usability**

These groups come from the two criteria of evaluation: document quality and usability. **Consistency** and **whitespace** pertain to document quality, **implementation errors**, **direct formatting controls versus preset style buttons**, and **usability** pertain to the prototype's usability.

The patterns are described with references to the tables. References are denoted as *Table name, Column name*. An occasional image or quote is shown for emphasis and clarity. Quotes are translated from Dutch by the authors. Translation and original are shown side-by-side.

Consistency The documents created by participant Robert and Timothy contain no consistency errors at all (Document structure analysis, column Markup error), a marked improvement from the previous test. Participant Eva's document contains four markup errors: where the implied level of a heading does not match the markup. She titled the document in a heading 1 style, rather than a title style. This error trickled downward unto other headings too, resulting in four mismatched headings (Document analysis stage II, column Markup error, also Eva's document stage II).

Whitespace The document analysis shows a drastic reduction of whitespace formatting as opposed to the previously produced documents (Document analysis, stage II). Only one redundant newline was found (in Eva's document stage II).

Implementation errors Two participants unwittingly encoded hard spaces in their HTML code (Document analysis stage II, column Editor error). All three participants inserted many newlines at the end of the document (Document analysis stage II, column Interface error). As argued in 4.3.3, this is due to a faulty implementation.

Direct formatting controls versus preset style buttons Eva, who formatted directly in her previous test, did so too in the prototype. Much to her frustration, she found that the bold/italic controls did not function as expected, and vice versa, that bold and italic were activated when changing only the font size (Observation analysis stage II, column Direct formatting, also Questionnaire stage II question A):

It is very annoying that changing the font size, bold and italic change too.	Het is heel vervelend dat als je het lettertype groter maakt bold en italic mee veranderen.

Interview with Louisa

The other two participants Robert and Timothy, who did use direct formatting controls previously, did not do so now. They immediately used the buttons for preset styles, rather than the direct formatting controls (Observation analysis stage II, column Preset styles). A reason for this difference cited by Timothy is that the prototype was simpler while still offering the features one needs in Microsoft Word:

It is simpler. The current version of Word has an information overflow: too crowded, too much functionality. But everything you need in Word is available.	[Het is] simpeler. Huidige versie van Word heeft een information overflow: te druk, te veel functies. Kunt er verder alles mee wat je in Word nodig hebt.

Post-task questionnaire Timothy

Usability Eva used and got frustrated with the abnormal bold and italic direct formatting controls. Her normalized SUS score was 51 (Questionnaire stage II). The other two participants did not use direct formatting and had significantly higher SUS score: 90 and 97 (Questionnaire stage II). This small test but large difference can be be an indicator of usability problems with the changed direct formatting behaviour and a satisfaction with a the simple buttons for preset styles.

4.3.5 *In summary*

Among the three writers, two formatting approaches were used. Eva used the Snap-to-Style direct formatting controls, while Robert and Timothy used the preset style buttons.

Eva's test showed usability problems with the bold and italic functionality of the modified direct formatting controls, specifically with the new bold and italic behaviour. Her document also displayed consistency errors. Robert and Timothy, on the other hand, immediately used the preset style buttons, where in previous tests both formatted directly. The reason for their change in behaviour is that the prototype's interface for preset styles was less cluttered and simpler than Microsoft Word. Their SUS scores report good usability and their documents display no consistency errors. The findings suggest that the current behaviour of the Snap-to-Style principle is poorly usable while the simplification of the preset style buttons is highly usable.

The use of newline formatting was reduced drastically among all participants, down to one occurrence.

Stage III: Prototype refinement and evaluation

The changed functionality of bold and italic seems to lead to confusion and frustration, the findings of the previous stage (4.3.4) suggest. Besides the poor usability, it also offers the writer no freedom of design. As Flynn (2014) points out as one of the main disadvantages of structured authoring, the use of a predetermined stylesheet precludes the writer from any design decision. Indeed, reasons cited in the first test for not using preset styles are that they simply do not look good (see 3.2.4)

However, we have also seen that a predetermined stylesheet removes unnecessary newlines. In this iteration, we refine the prototype to retain the mechanism that eliminates unnecessary newlines, while improving the behaviour of the bold and italic buttons, and offering users the some freedom of choice in design.

Thus in the next iteration of the Snap-to-Style prototype, we repurpose the bold and italic buttons. Instead of involving them in the snapping mechanism, we allow the writers themselves to assign weights and styles to headings, and thereby defining the style for that particular heading. All styles are initially of regular weight (not bold) and of roman style (not italic), until changed by the writer. So, for example, if a Heading 1 is currently not bold, but then made bold by use of the bold button, we update *all* Headings 1 to become bold. The font sizes retain their behaviour of "snapping" to headings, as they did previously.

We also introduce a font-family picker dropdown menu. In this test, only two font families are included: a sans-serif (Calibri) and a serif (Cambria). These were the two fonts found in the documents produced in the very first test, so we may assume the writers are familiar with them. When changing the font-family, the entire document is updated. The reasoning is twofold: the documents produced in the very first test are all set in one family, and professional typographers agree that it is rarely necessary to mix fonts within one document (Bringhurst, 2015; Butterick, 2010).

With these updates the entire document still has consistent formatting, and the underlying document is still marked up descriptively, but the user is allowed the freedom of design. In the figures below, the new snapping mechanism is illustrated diagrammatically. Note its strong similarity to the previous diagram (font weight and style controls are removed). Secondly, a diagrammatic overview is given for changing the font weight and size (in this case for Heading 1, but it of course applies to every heading). Keep in mind that when altering one heading's weight and style, all headings of the same level are updated to match the style.

Finally, note that the interface still looks exactly the same, save for the addition of the font-family picker dropdown menu. It is available at daniel.roeven.com/snap-to-style-2.

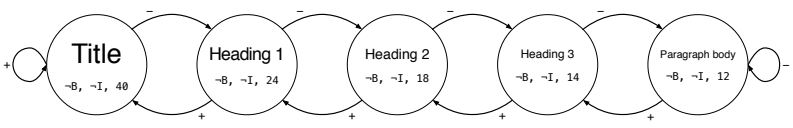

Figure 5.1: diagram of font size actions and their corresponding preset style

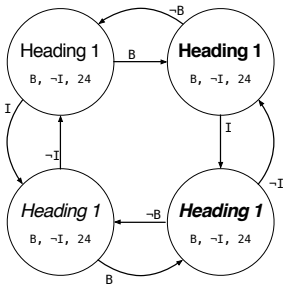

Figure 5.2: diagram of font weight and style formatting actions

5.1 METHODOLOGY

The prototype was evaluated on quality of document formatting and usability. It was done in the same manner as the previous evaluation (see 4.2): a moderated task test, a post-test questionnaire, and a document analysis. A new handwritten document was created (see appendix C1), but the test and the questionnaire remained the same. Again, the test was conducted with three participants. Robert and Timothy were selected to see if they would make use of the new freedom of design (having seen the interface before). For Louisa the interface was new.

The data analysis of the three sources of data (observations, questionnaires, documents) took place in the same manner as in stage II with the addition of a few new categories of codes. These are described in the results.

5.2 RESULTS

The coding or analysis of each of the three sources of data is presented in a table, included in the appendix (C5-C7).

Then we present our findings, combining the information from the three data sources, grouped by subject.

5.2.1 *Moderated task test*

The result of the coding of the observations are presented in table *Observation analysis stage III*. The same codes were used and the same categories are displayed in the table.

5.2.2 *Post-task questionnaire*

The questionnaire was summarized in table *Questionnaire stage III*, comments for particular questions included, and a normalized SUS score was calculated.

5.2.3 *Produced documents*

The produced documents were analysed for errors. That is summarized in *Document analysis stage III*. The produced documents were annotated in the same manner as in the previous stage, with the addition of a few new categories of errors:

- *Editor error*
  - *Tab*
  - *Newline*
  - *Inline styles*
- *Interface error*
  - *Unresponsive*

Tab editor errors are where the prototype stores a tabs as hardcoded tabs in the HTML output. This is a bug and should be removed. *Newline editor errors* occur when two seperate lines are selected and one style is applied. The editor then formats this as <h1> Foo
 Bar </h1> when it should be formatted as <h1> Foo </h1> <h1> Bar </h1>. This is faulty behaviour that should be removed. *Inline styles* are a remnant of the prototype's under-the-hood WYSIWYG engine. It previously stored procedural markup in the HTML. Inline styles are when, erroneously, procedural markup is still stored. The descriptive markup is correct in this case, but is overridden visually by faulty procedural markup. It is a bug and should be solved. *Unresponsive interface errors* are when the interface displays faulty behaviour. The formatting buttons become unresponsive. The "complete" button still works and HTML can be generated.

5.2.4 *Findings*

The three tables then display information on usage of and problems with the prototype. Findings that emerge from frequent entries in the tables are then described, grouped by subject:

- **Consistency**
- **Implementation errors**
- **Whitespace**
- **Freedom of design**
- **Style design**
- **Direct formatting controls versus preset style buttons**
- **Usability**

These groups are the same as in stage III: they pertain to the two evaluation criteria document quality and usability. Two new groups were introduced: **Freedom of design** and **style design**. These groups contain findings pertaining to the change that was made in this refinement.

The findings are described with references to the tables. References are denoted as *Table name, Column name*. An occasional image or quote is shown for emphasis and clarity. Quotes are translated from Dutch by the authors. Translation and original are shown side-by-side.

Consistency Robert and Timothy's documents displayed no consistency errors. Louisa's document displayed two markup errors (Document analysis, column Markup error).

Editor and interface errors Similar to the previous test, there were some hardcoded spaces and a hardcoded break in the output HTML (Document analysis stage III, column Editor error). There were also some newlines at the end of the document (just outside the viewport), similar to the previous test (Document analysis stage III, column Interface error). In Louisa's test, the interface became unresponsive (Document analysis stage III, column Notes). HTML output was still possible, and the document was half complete. This half was analysed in the same manner as the others.

Whitespace One extraneous newline was introduced (Document analysis stage III, column Formatting error).

Freedom of design Louisa, for whom this was the first time working with the prototype, noted that there were fewer choices then in Microsoft Word. She appreciated the limited palette of options and likened it to the

difference between the two mobile operating systems Android and iOS (Questionnaire stage III, question A):

> It reminds me of the difference between Android and iOS: lots of choices and lots to go awry, versus few adjustments, little to goes awry.
> Post-task questionnaire Louisa
>
> [Het doet me denken aan het] verschil tussen Android en iOS: veel keuze, veel fout gaan, vs. weinig aanpassen, weinig fout gaan.

Robert agrees with this simplicity (Questionnaire stage III, question A), and lists it as the reason he uses the preset style buttons.

Timothy, however, disliked the lack of complete freedom. Or rather, the increase in freedom (font family, bold, italic), which made him notice the absence of complete freedom. When design the styles, there were not enough options he could change. In the first test, he trusted the stylesheet designers's choices (in this case us, the authors). When left to his own devices for design, he found a lack of freedom bothering him (Questionnaire stage III, question 1 and A):

> In the previous prototype I placed trust in the pre-made styles. Now I have too little options. Too bad the font-size couldn't be changed. [...] This is more like a restricted version of Word, where the freedom Word offers is missed.
>
> Post-task questionnaire Timothy
>
> [Daar had ik] vertrouwen in voorbepaalde stijl. Hierbij is keuze te klein. Jammer dat fontsize niet veranderd kan worden. [...] Dit lijkt meer op een beperkte versie van Word, waarbij de vrijheid die Word biedt gemist word.

Style design Timothy and Robert styled their documents with the bold and italic buttons. Louisa only changed the font-family, leaving all headings in regular weight and roman style (Document analysis, stage III, also documents from each participant).

Direct formatting controls versus preset style buttons All participants used the preset styles buttons. Louisa began formatting with the direct formatting controls for increasing and decreasing font size, but then switched to using the preset style buttons.

Usability All normalized SUS scores were high ranges: 91 (Louisa), 93 (Robert) and 83 (Timothy). Timothy's score decreased a little from the previous test for reasons described above (insight into lack of freedom).

Stage IV: Evaluation of the direct formatting controls

6

In the previous tests, nearly all participants gravitated towards the use of preset style buttons. We have thus found one way of applying the benefits of structured documents to WYSIWYG environments: stimulate the use of preset style buttons.

We identify three reasons for the switch to preset style buttons. The first was directly given by two participants (Robert and Timothy): simplicity. The interface is less cluttered and more limited, while still offering the necessary functionality. By making the preset style buttons less intimidating, their usage increases.

The second reason is a matter of aesthetics. In stage I we have seen that writers purposefully refrain from using preset styles simply because they do not look good. This reason resurfaces in stage II and III: in stage II, Timothy had confidence in the stylesheet designer's choices, and appreciated it's limited palette of options. In the stage III, where a large portion of aesthetics was up to the writers, his confidence faltered.

The third reason came forth from Louisa's test: she began formatting with the direct formatting controls but then switched to the preset style buttons. When formatting directly, the style button for the currently active style is enabled. Changing the currently active style, then changes the enabled style button changes. Thus, it is possible that the changing of the enabled button acted as a visual stimulus allowing for the insight that the two ways of formatting actually do the same thing. With that insight, the preset style buttons are the most straightforward way of formatting.

Now that this approach of stimulating preset styles has been proven to be effective (either via simplification, via improved style design, or via a visual stimulus), the focus returns to the direct formatting controls. They have not yet been sufficiently evaluated, as all participants thus far made use of the preset style buttons. We will therefore evaluate the Snap-to-Style direct formatting controls in this stage.

We made no changes to the implementation of the direct formatting controls. They retained the same functionality as in stage III. However, the preset style buttons were removed. We thus removed the possibility of writers shifting to the preset style buttons. In order to specifically test the effectiveness of the Snap-to-Style direct formatting controls, we eliminated the possibility of participants shifting to the use of preset style buttons.

It is important to note that this is not a suggestion for an actual approach; a final design should most definitely include both. A shift in usage from direct formatting towards preset styles is a valid and commendable phe-

nomenon. But to specifically test the direct formatting controls we eliminate the possibility of this shift.

6.1 METHODOLOGY

The prototype was evaluated on quality of document formatting and usability. It was done in the same manner as the evaluation of stage II (see 4.2): a moderated task test, a post-test questionnaire and a document analysis. The handwritten document used in stage III was recreated (the document was previously unseen by the participants of this round). The test and the questionnaire remained the same.

The test was conducted with two participants. Participant Adam was selected for two reasons. In stage I he formatted directly, so it is likely he would again in this test. He has also not seen the prototype before, so he has not seen the preset style buttons. He did not yet shift towards the preset style buttons, so we do not "remove" formatting options. Testing with Robert or Timothy while removing the style buttons, to which they had just accustomed, would make for poor usability. The other participant is Eva. She has tested a previous version of the Snap-to-Style prototype in stage II. There, she was the only participant who made use of the direct formatting controls. As their functionality has changed a bit (see the changes made in stage III), it is worth evaluating them again on a participant who uses direct formatting controls.

The test was doubly recorded: via a recording of the screen and via a camera aimed at the screen and keyboard. Observations are then made with the recordings. This allows for a greater observational precision.

The data analysis of the three sources of data (observations, questionnaires, documents) took place in the same manner as in stage II with the addition of a new category of error. This is described in the results.

6.2 RESULTS

The coding or analysis of each of the three sources of data is presented in a table, included in the appendix.

The observations were coded and are presented into table *Observation analysis stage IV*. The questionnaire was summarized in table *Questionnaire stage IV*, comments for particular questions included, and a normalized SUS score was calculated.

The produced documents were analysed for errors. One new category was added to the document analysis: space formatting errors. This is a form of whitespace formatting: spaces are added to the beginning of a paragraph, as a form of indentation. These spaces are hardcoded in the

output. The document analysis is summarized in *Document analysis stage IV*. These tables are available in appendix D3-D5.

These three tables then display information on usage of and problems with the prototype. Findings that emerge from frequent entries in the tables are then described, grouped by categories:

- **Consistency**
- **Implementation errors**
- **Whitespace**
- **Style design**
- **Usability**

These categories are similar to the categories from stage III: they pertain to the two evaluation criteria document quality and usability.

The findings are described with references to the tables. References are denoted as *Table name, Column name*. An occasional image or quote is shown for emphasis and clarity.

6.3 FINDINGS

Consistency The produced documents are not marked up correctly (Document analysis stage IV, column Markup errors). Both participants did not mark up the title as <h1> but as <h2>. Then, the offset error trickled down, with the result that every subsequent heading is also marked up incorrectly (see produced documents). In Adam's case this is due to him not being able to select fontsize 40 (which corresponds to the title's markup tag). This fontsize was unavailable because a popover covered the option in the dropdown menu. See figure 6.1 for clarification. He then set every heading with an offset of one (Heading 2 in <h3>, Heading 3 in <h4>).

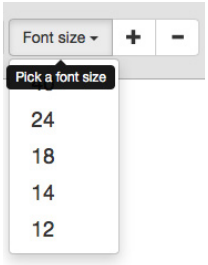

Figure 6.1: popover covers top fontsize option.

Implementation errors Spaces were hardcoded in the output HTML (Document analysis stage IV, column Editor errors, also produced documents).

Whitespace Eva twice added an unnecessary newline. Adam used spaces to indent every second paragraph (Document analysis stage IV). See figure 6.2.

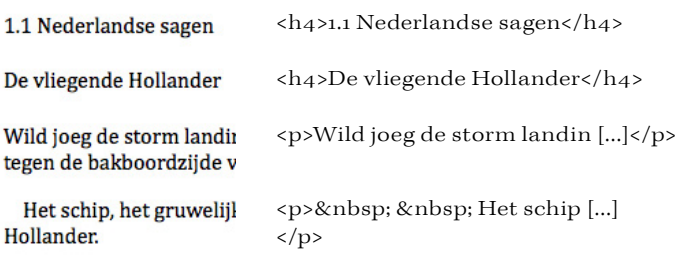

Figure 6.2: document and HTML code. Indent made with   in the second paragraph.

Style design Both participants's documents display no bold or italic fonts (Document analysis stage IV, column styling, also produced documents). Eva first wrote the entire text, then added formatting afterwards (Observation analysis stage IV, column usage). When formatting, she first applied bold, and then increased the font size. Because the font size was not increased yet, the document element was marked up as a paragraph. Applying bold to a paragraph element makes all paragraphs bold. This resulted in her entire document becoming bold (Observation analysis column Direct formatting). She undid the bold, thereby undoing bold in the whole document, and consequently did not use bold or italic anymore.

Usability The normalized SUS scores are 85 for Adam and 63 for Eva. Adam appreciated that one needs not select the entire document element to change its properties, but placing the cursor inside the document element was enough to apply the formatting to the entire element (Questionnaire stage IV, question A, B). For example, when increasing a heading's font size, one does not need to select the entire element, as long as the cursor is placed inside the elements's text. This is because the style is changed for the element, rather than for the text, meaning that the formatting is consistent and thus applies to the entire element.

Conclusions and discussion

7

7.1 CONCLUSIONS

We first answer the two subquestions. The subquestion corresponding to the first stage of our research is: *how do writers currently work with WYSIWYG word processors?* We then answer the second subquestion based on the second, third, and fourth stage in our research: *does combining the knowledge of structured documents and the insights into WYSIWYG editing lead to improved usability and document formatting?* These conclusions then allow us to answer the main research question: *how can a WYSIWYG word processor be designed to incorporate the benefits of structured documents?*

7.1.1 *How writers currently work*

Many writers format directly and use newlines as whitespace formatting. The documents created by these participants display typographical consistency errors, where headings of the same level are not typographically equal, or where headings of the different levels are typographically equal.

The writer who formatted with preset styles produced a document free of unnecessary newlines and consistency errors.

Reasons for not using preset styles are that writers do not know how they work and that they do not suffice aesthetically: preset styles do not look good.

Participants experience frustration when layout suddenly changes. This is partly due to newline formatting.

7.1.2 *Combining the knowledge of structured documents and the insights into WYSIWYG editing*

Preset styles offer the benefits of structured documents in WYSIWYG environments. They are a WYSIWYG version of descriptive markup. They ensure consistency, which we have seen is a source of formatting troubles.

Preset styles allow for cross-referencing. This means that a table of contents (TOC) can be generated automatically. We have seen that the writer who formats with preset styles does automatically generate a TOC, while writers who do not write the TOC by hand. Automatic generation is quicker, easier, and guaranteed to be correct.

Reducing newline formatting reduces suddenly-changing layout. With properly designed preset styles, appropriate whitespace is included in the styles as margins. Manually adding whitespace by inserting newlines is then no longer necessary.

7.1.3 *Designing a WYSIWYG word processor to incorporate the benefits of structured documents*

Not all writers use preset styles. Many writers format directly. Improving usability and document formatting can be done in two ways: encouraging the use of preset styles and repurposing direct formatting controls to display the functionality as preset styles.

Encouraging the use of preset styles can again be done via two ways: simplifying the relevant affordances (preset style buttons) and improving the preset style aesthetics. The first can be done by offering fewer styles and making the interface less cluttered. These approaches have been evaluated. Two participants switched from using direct formatting in WYSIWYG processors to preset styles in our prototype. Their documents are free of consistency errors. SUS scores indicate good usability.

Mapping direct formatting controls to preset styles can be done as follows: limit the font size affordances (dropdown menu, larger, and smaller buttons) to predetermined sizes, with each size corresponding to a style (title, headings, paragraph). The editing software then keeps track of the style, inferred from the font size, by adding invisible descriptive markup to the relevant document element. Style design can then be done by writers. Whenever a bold or italic is applied to a style, the editing software updates the stylesheet, so that formatting remains consistent.

This approach is promising but still has some teething problems. Consistency errors can occur if styles are not used sequentially: if a document element is not set in the correct size (thus invisibly marked up with the incorrect descriptive tags), an offset error trickles downward upon all other elements too. For example, Headings 2 are tagged as `h3`, Headings 3 as `h4`, and so forth. Usability errors occur when writers misinterpret the changed direct formatting controls.

Finally, using either preset styles or the repurposed direct formatting controls clearly reduces unnecessary newlines.

7.2 DISCUSSION

The finding that only few writers make use of preset styles is consistent with research by (Sørgaard & Sandahl, 1997). Despite their findings being almost 20 years old, it seems the low adoption rate has not changed. Results from this thesis indicate that simplifying the preset styles and increasing their aesthetic appeal can improve the adoption rate.

A third way of increasing preset style usage identified but not confirmed in this thesis is described in stage IV: the visual stimulus of the changing style buttons. When increasing or decreasing font size, the preset style

buttons change accordingly. This acts as a visual cue inviting the writers to recognize the styles and direct formatting as doing the same thing.

In a large research identifying ways of improving structured authoring environments, Flynn (2014, p. 369) describes WYSIWYG environments producing HTML5 that recognize document structure based on typographic information as leading the next phase of development in word processing. This thesis was in ways the opposite: it identifies ways of improving WYSIWYG environment by taking cues from structured authoring environments. Nevertheless, results are similar: visual editors producing HTML5 (or any structured document, but HTML5 specifically due to its web- and ebook- related capabilities) that infer structure from the writers's current way of working are usable and produce high-quality documents provide good usability and high-quality documents.

7.2.1 *Limitations of the study*

The primary concern of this research is that the direct formatting controls has been evaluated with only two participants. Although valid usability problems have been found, there is no guarantee that the found problems are indeed the majority of usability problems (stage IV).

The sample size is relevant in the SUS scoring also. Usability experts have found that five participants are enough to generate precise results for SUS scoring (Sauro, 2013). However, these five participants used different formatting approaches in the same task. In the prototype, three participants formatted using preset style buttons, while two others used the direct formatting controls. As a whole, we can say that the prototype performs rather well on the SUS test. The results do not become less reliable at smaller sample sizes (Sauro, 2011), but smaller sample sizes give do poorer estimates of variability. This is important when attributing these scores to the approaches separately, because the sample size decreases.

It is also important to note that the participants were all acquaintances of the author. We have to consider the possibility that the participants performed socially desirable in the tests and questionnaires. This could have happened in two ways: they tried their best to format correctly or gave desirable answers when speaking of the software in the interview. The former was combatted by asking them to format the same as they would when writing a formal document for their education (that way the "correct" way of formatting is their usual way). The latter was combatted by using the standardized SUS test, and informing the participants that it is the software that is being tested, not the participant's intelligence. Because the SUS test uses a Likert scale, the possibility is reduced of the participants binarily "agreeing" that the usability is good. By stating that

it is the software being tested rather than the participant, the threshold for offering critique is reduced, because participants no longer fear for judgement that "they do not get it".

A few remarks can be made on the switch that two participants made from direct formatting to preset styles. The first is that a large portion of the interview in stage one revolved around the manner of formatting. Participants were questioned on their use of direct formatting and the why-not of preset styles. It is possible that because of this focus, these participants switched to the use of preset styles in later stages. It is also possible that in the prototype it felt "safer" to explore and try a different formatting approach. Because it is just a test and did not have any consequences for the participants outside of the test, whereas writing formal documents for their education does. In that case it is understandable that in the test participants tried something new, whereas when writing a report or essay they stick to what they know.

Finally we mention that in stage one, due to time constraints, the interview with Adam was conducted digitally: questions were emailed to him which he then answered. Also, in stage IV, near the end of the test, he was explicitly encouraged to add formatting, because when he indicated that when he was finished, no formatting was added at all.

7.2.2 *Future work*

There are two principal areas that require further research. The repurposed direct formatting mechanism will need to be further evaluated on usability. It has only been properly tested with two participants, due to other writers shifting to the use of preset style buttons. Results indicate that there are usability problems. One of these problems is that the preset style has to be applied (by setting the font size) before style design (setting bold or italic). Doing it in this order allows the environment to keep track of descriptive markup, and maintain a stylesheet for the styles. Doing it the other way around leads to the incorrect styles being changed. Flynn (2014) proposes a change to the bold and italic buttons: rather than being toggled on/off, they query the user for intent upon being pressed ("are you applying italic for aesthetics or emphasis?"). Such queries are then used to determine wether to update the stylesheet or update the markup. Perhaps they could also be used to remove the encountered problems: "are you applying italic to paragraphs or to headings?".

The second area is where a mismatch of markup and implied headings exist. We have seen that an offset error occurs when one style is set in the wrong preset style. Every subsequent heading is then also set in the wrong style. A possible solution for this is to set the largest document element to

Heading 1. Then successively set each smaller element to a smaller style. Remapping the preset styles to the selected font sizes for each document, rather than having a predetermined mapping for all documents.

Expanding on this idea, Fuß, Gatzemeier, Kirchhof, and Meyer (2004) have done research on inferring structure from typography. This goes further than just addressing font size. They propose a system which infers document structure from applied procedural markup. A limitation of their system is that the procedural markup needs to be entirely consistent. We have seen that this is not always the case. That is why such a system could perhaps be combined with the repurposed direct formatting controls as we have seen them in this thesis. The off-by-one markup errors that our mechanism induces can be perfectly solved by the system proposed by Fuß et al. (2004).

References

Bringhurst, R. (2015). *The elements of typographic style* (4.1 ed.). Hartley & Marks, Publishers.

Brooke, J. (1996). SUS: a "quick and dirty" usability scale. In P. W. Jordan, B. Thomas, B. A. Weerdmeester, & A. L. McClelland., *Usability evaluation in industry*. Taylor & Francis.

Butterick, M. (2010). *Butterick's practical typography*. Retrieved 4 Nov 2015, from http://practicaltypography.com/mixing-fonts.html

Chamberlin, D. D., Hasselmeier, H. F., & Paris, D. P. (1988). Defining document styles for WYSIWYG processing. In *Proceedings of the international conference on electronic publishing on document manipulation and typography* (pp. 121–137). New York, NY, USA: Cambridge University Press. Retrieved from http://dl.acm.org/citation.cfm?id=51292.51302

Coombs, J. H., Renear, A. H., & DeRose, S. J. (1987, November). Markup systems and the future of scholarly text processing. *Communications of the ACM, 30* (11), 933–947. Retrieved from http://doi.acm.org/10.1145/32206.32209

CTAN: TeX. (2016). Retrieved 11 Jan 2016, from https://www.ctan.org/tex

Ergonomics of human-system interaction – part 210: Human-centred design for interactive systems. (2010). Retrieved from http://www.iso.org/iso/catalogue_detail.htm?csnumber=52075

Flynn, P. (2014). *Human interfaces to structured documents*. Unpublished doctoral dissertation, National University of Ireland, Cork.

Fuß, C., Gatzemeier, F., Kirchhof. M., & Meyer, O. (2004). Inferring structure information from typography. In P. King & E. Munson (Eds.), *Digital documents: Systems and principles* (Vol. 2023, p. 44-55). Springer Berlin Heidelberg. Retrieved from http://dx.doi.org/10.1007/978-3-540-39916-2_4

Gócza, Z. (2010). *Myth 21: People can tell you what they want*. Retrieved 4 Nov 2015, from http://uxmyths.com/post/746610684/myth-21-people-can-tell-you-what-they-want

Johnson, J., & Beach, R. J. (1988, January). Styles in document editing systems. *Computer, 21* (1), 32–43. Retrieved from http://dx.doi.org/10.1109/2.222115

Lamport, L. (1986). *LaTeX: A document preparation system*. Addison-Wesley.

Mathis, L. (2011). *Designed for use: Create usable interfaces for applications and the web* (1st ed.). The Pragmatic Bookshelf.

Nielsen, J. (2000). *Why you only need to test with 5 users*. Retrieved 29 Dec 2015 from https://www.nngroup.com/articles/why-you-only-need-to-test -with-5-users

Pronin, E. (2009). Chapter 1 the introspection illusion. In *Advances in experimental social psychology* (pp. 1–67). Elsevier BV. Retrieved from http://dx.doi .org/10.1016/S0065-2601(08)00401-2

Reid, B. K. (1980). A high-level approach to computer document formatting. In *Proceedings of the 7th ACM sigplan-sigact symposium on principles of programming languages* (pp. 24–31). New York, NY, USA: ACM. Retrieved from http://doi.acm.org/10.1145/567446.567449

Roberts, T. L. (1980). *Evaluation of computer text editors*. Unpublished doctoral dissertation, Stanford University, Stanford, CA, USA.

Sauro, J. (2011). *Measuring usability with the system usability scale (SUS): MeasuringU*. Retrieved 15 Jan 2016, from http://www.measuringu.com/sus.php

Sauro, J. (2013). *10 things to know about the system usability scale (SUS): MeasuringU*. Retrieved 15 Jan 2016, from http://www.measuringu.com/blog/10-things-sus.php

Shneiderman, B. (1983, August). Direct manipulation: A step beyond programming languages. *Computer, 16* (8), 57–69. Retrieved from http://dx.doi.org/10.1109/MC.1983.1654471

Sørgaard, P., & Sandahl, T. I. (1997, Jan). Problems with styles in word processing: a weak foundation for electronic publishing with SGML. In *Proceedings of the thirtieth hawaii international conference on system sciences* (Vol. 6, p. 137- 146).

Style basics in word. (n.d.). Retrieved 11 Jan 2016, from https://support.office.com/en-us/article/Style-basics-in-Word-d382f84d -5c38-4444-98a5-9cbb6ede1ba4

Summernote - super simple WYSIWYG editor on bootstrap. (2015). Retrieved 4 Nov 2015, from http://summernote.org

Afterword

Humans have been writing for an astoundingly long time. Recorded history, by its very definition, begins when the written word first appears. Some of the oldest pieces of writing known to mankind date back to 3500 BC. Ever since, impossibly intricate writing has been carved with chisels and written with feathers, on media as diverse as stone or wax, parchment or papyrus. Generations of lettercutters and stone carvers, calligraphers and master scribes have upheld the wondrous tradition.

It is to inventors Johannes Gutenberg and the lesser known Bi Sheng (畢昇) that we owe the marvellous movable type, which allowed for the large-scale reproduction of writing. Enabled by this device, Erasmus and Luther sold hundreds of thousands of copies of their books. Unthinkable! — for men and women of the fifteenth century.

But it is not until the 1860s that reliable and standardised writing became truly available to the public: the introduction of the typewriter. Masterpieces of meticulous engineering, and still much beloved pieces of design today. They proved incredibly useful and were used by practically all writers for well over a century. Their reign came to an end in the late 1980s, when personal computers became commonplace.

The word processor is the modern-day equivalent of the seemingly indispensable typewriter. Word processors are used by schoolchildren and poets alike. Indeed, many editors suiting many different writers have gained foothold in the modern writing world.

And so the delightful art of writing —which makes us so uniquely human— has been upheld by the scribes and carvers; the inventors and operators of the printing press; the engineers of the typewriter. And now, that torch is passed to the designers of software. It is up to us to ensure it thrives in this digital age.

Appendices not included.