What you see is Wiki - Questioning WYSIWYG in the Internet Age

by Christoph Sauer, 5. August 2006

Almost 20 years ago, Leslie Lamport wrote about the advantages of logical document production over visual document production [Lamport 87]. Until recently it seemed that his insights in the disadvantages of visual text production with so called WYSIWYG editors seemed almost forgotten, because of the ubiquity of programs like Word. Only a small group of people, mainly in research, still used the logical approach to document production by using his LaTex program, appreciating the ability to concentrate on the content and leave the formatting to the program--until recently. With the success of editable web pages by the advent of wikis and its most popular application, Wikipedia, logical text production as Lesley Lamport described it, has found its way into the mainstream. There is a long standing discussion in the wiki world on whether to move back to visual text production in wikis again, since end users who only know WYSIWYG editors are suddenly confronted with wiki markup which they find disturbing at first.

In the rest of this text, I will explore this discussion and present the pros and cons of wiki markup and show why wiki markup is useful to keep as part of a grammar for online cooperation. I will make the statement that WYSIWYG in the internet age is not useful anymore. Still many users consider the current way of editing a wiki old-fashioned and demand a modern way of editing pages. I will show a way out of the dilemma by introducing a proof of concept editor called "WikiWizard" that implements a new approach to wiki editing by mixing wiki markup with visual feedback using elements of advanced markup highlighting and wizard dialogs to ease logical editing.

Are Wikis Usable?

When editing wikis you only have a simple text editor, that creates no visual feedback on how the end result might look like. You are limited to use the characters available on your keyboard to enter text. In order to create formatting and structure in wikis, you use characters preceding or embracing your text. This notion is not new. In HTML you use those character sequences to "mark up" text, like headings and links. In contrast to HTML however, the character sequences to markup text are any simpler, designed to keep the text readable. For users there is much less markup to remember than in HTML. In most wikis there are only 5 to 7 elements to remember for the average user:

Basic	**JSPWiki Markup**
Headings	`!Small, !!Medium, !!!Large`
Bold and Italic	`__bold__, ''italic''`
Links	`Internal Link: [Wikipage], [External Link| http://www.wikipedia.org]`
Lists	`* Bullet List Item, # Numbered List Item`
Paragraphs	`empty line seperates paragraphs, forced line break is \\`
Advanced	**JSPWiki Markup**
Attachments	`Inlined Image: [landscape.jpg], link to an attached [word.doc]`
Tables	`|column 1| column 2`

Basic Elements of Wiki Markup

In this sense wiki markup is a much simplified version of HTML, as HTML was a simplified version of its predecessor SGML. The simplification process of text markup has therefore reached a point where you can't make it much simpler; and as it turns out, it seems to be finally simple enough for end users. A Canadian study among 4th grade students (age 8 to 9) showed that indeed wiki markup can even be used by children after a 15 minute introduction [Désilets 06]. The study shows that the wiki markup itself is by far not the biggest usability issue with wikis. The real issues that were revealed by this study are the understanding of the hypertext nature of the Internet, and related to that, link creation and management. The success of wikis and Wikipedia in particular therefore indicates that we seem to have the missing simple markup finally to get the masses involved. Although wiki markup can be used by the general public, it is still a usability issue. What are the reasons for that?

Wiki Markup Cons

There are several reasons why wiki markup is a usability problem; some are real issues that have to be coped with by better editors and improvements in wiki engines. Others are merely to the fact that users only know the visual approach right now. Users are not aware of the advantages of the logical approach because there is hardly any discussion about it outside the academic world. Here are the basic usability issues, where if appropriate I will already give some hints on how to cope with them.

This is not for me: People are puzzled when they first see a text with wiki markup. WYSIWYG editors are well known today and their usage is even taught in school. After the first introduction, the average user seems to be using an explorative approach to get used to a new program. Programs like Word let users click buttons and search for menus to achieve certain results like making a word bold. In a simple text editor using wiki markup, however, a user has to learn wiki markup first, or be introduced by someone. Consumers often react with a “This is not for me” attitude to this, even if it just takes 15 minutes, because the advantages of logical text production are not obvious to them at first.

People want to change the layout: Users miss their ability to format freely. People don't like to be limited, especially because you take a freedom away from them they previously had in WYSIWYG editors. In order to give back this freedom to end users, we have to give them the ability to change the layout too. But it has to be made a separate application, that might be limited to a certain group of people - designers. This application should use the WYSIWYG approach instead of presenting the end user with CSS files. In case of a personal wiki the user should have access to this WYSIWYG application which does not have a primary goal of editing but design.

Losing the overview: This is an issue that even proponents of wiki markup face when using a simple text area with no visual feedback. Navigation is hard in a sea of monospace letters. This issue has partly been adressed by the ability to edit only parts of a document (e.g. links for editing behind every heading in Wikipedia). Still the monospaced font makes it harder to read a document in its source view, which feels like text processing of the „old days“. This issue has to be addressed by advanced markup editors.

Wiki markup mess: This is a serious homebrew problem of the wiki community. There are numerous popular wiki engines out there, and every wiki engine uses its own wiki markup. This is also known as the *wiki markup mess*. All attempts to standardize wiki markup have failed so far. This makes it really hard for users to edit different wikis that run on different wiki engines. The wiki community has to get on the wiki markup mess with a common wiki markup standard to cope with that problem.

If there are so many usability obstacles, why should we not simply deploy WYSIWYG editors in wikis to solve those problems? In order to answer this question we have to look for what purposes WYSIWYG editors where originally designed and what problems they have online.

WYSIWYG in the Internet Age

In a world where we can always be online, we can accept that paper is no longer the primary output medium. Current WYSIWYG editors that mix content with layout do, however, target this medium historically and conceptually. We are now at a stage where we could overcome the limitations of the physical world, but we still use the metaphors of the offline era to create documents, and we increasingly sense the difficulties that are involved with those approaches. The author of a document has to be prepared so his text will be published in different layouts in the future.

Think of the example of email: if you format your email on your email client software, you cannot expect that the recipient on the other side uses the same email client or the same settings. He might have turned HTML off for security reasons. Therefore what you see is not what your recipient gets. The same is true for instant messaging. When editing webpages you can not expect that the layout stays the same over time, because you might refresh the looks from time to time. New target groups may even force you to have different views of different representations.

Not only has the medium changed, but also our habits are changing how we consume and produce content. More and more people become knowledge workers that surf the web to gather information and remix text to create new documents containing information, ideas and concepts at a rapidly growing rate. Those knowledge workers, bloggers for example, often copy and paste code. If you write a larger document in this way from several authors you might have to reformat everything, and you will have to spend a lot of time doing this. Those users are however concentrating on the content and have no time to care for layout while writing. WYSIWYG is not useful for them anymore.

Wiki Markup Pros

The logical approach would therefore yield huge benefits over the visual approach in the internet age. Times have changed since the first introduction of visual editors. Here are reasons why we should **NOT** give up on wiki markup in favor of WYSIWYG editors:

Concentration on the content: The writer can concentrate on the content. It comes as a relief to authors that they no longer have to care for the layout, especially in large documents like technical books and articles. What the author sees is what he means, not what he gets. He can leave the formatting to the machine, that can now care for a context sensitive display. He also no longer prone to typographical errors.

Context sensitive display: No matter which medium is used to display the text, the layout can change accordingly, defined by the designers of this medium. Webmasters can enforce a common look and feel in an installation with many users. This creates an homogenous design that improves the readability and acceptance of a webpage. The reader can expect an optimized view for the medium and device he is currently using, for example a printout, a handheld device. This is especially a concern among the disabled.

Speed: Once you have learned wiki markup you don't need toolbars and dialogs anymore. Especially the easy embedding of linking which is an essential part of web pages does not need an extra call to a dialog box and is obvious in the text itself. The wiki markup becomes the *User Interface* which stands for itself. For knowledge workers like bloggers, fast editing becomes essential in order to not lose the *flow* in information hunting - collecting links and adding their thoughts to them in a blog entry.

Simplicity: Imagine a world where basic computer literacy would only require knowledge of pages interlinked with each other, and in order to edit those pages you have to remember seven short character sequences. Applications to edit are everywhere - because it is the simplest thing that could possibly work, like Ward Cunningham puts it [Cunningham 04], wiki markup would run on almost any device already out there. You can even create a document with something as basic as a cell

phone text message. This also means a productivity gain for the software industry, because a wiki markup standard would provide a simpler abstraction layer than HTML. Wiki Markup could also be used to simplify dynamic web page programming by replacing HTML as the primary output format. Therefore it allows easy evolving of wiki engines and web applications across the board.

Concentrating on the content and the ability to create context sensitive display could be summed up in the advantage of the separation from model and view. This is the main advantage of logical document production that would alone justify the addition of a new alternative to the old paradigm of WYSIWYG editors. Let me explain this concept in more detail.

The Essence of a Document

One year before Lesley Lamport in 1986, Fred Brooks published his well know essay called "There is no Silver bullet: Accidents and Essence of Software engineering" [Brooks 86]. He predicted that we would see no advances of scale in programming until we separate accidental task from the essential tasks - and the essence is, as he writes, *what is the same in many different representations*. If we transfer this to web authoring the accidental tasks are the format, because it will change in different representations. Whereas the essence, that is the same in many different representations, is the content itself plus its emphasis through structure and references. Structure is represented by headings, bold, italics, etc. while links to other documents and web pages make up the references. This means that wiki markup is part of the essence of a document--what the author wanted to preserve as a document in order to convey his ideas to the reader.

As a design pattern in software development we know this notion as model view pattern. People that edit in a WYSIWYG editor do not sense this distinction. What we need is a new computer literacy where people are aware of this distinction. Teaching people the difference between model and view with the help of wiki markup in their basic education would produce huge productivity gains in document production. Wiki markup should be seen as part of the grammar for online authoring instead of hiding it behind a certain view. If the exclamation point (!) at the end of a sentence is part of the emphasis of an offline grammar of our written languages on paper, the exclamation point at the beginning of a sentence, producing a first level heading, would be the grammar of online cooperation, where the grammar of the offline era is a subset of it.

Advanced Markup Editor Wiki Wizard

Still users demand a rich user experience. Current simple text editors would not be enough to replace the current visual WYSIWYG editors. Issues like the lack of explorative usage and losing the overview in a sea of monospaced letters have to be addressed by new editors. This could be solved by editors that combine the relatively simple interface of word processors with the concept of source code highlighting, which programmers know from development environments. Combining these features would create something that looks more familiar to end users.

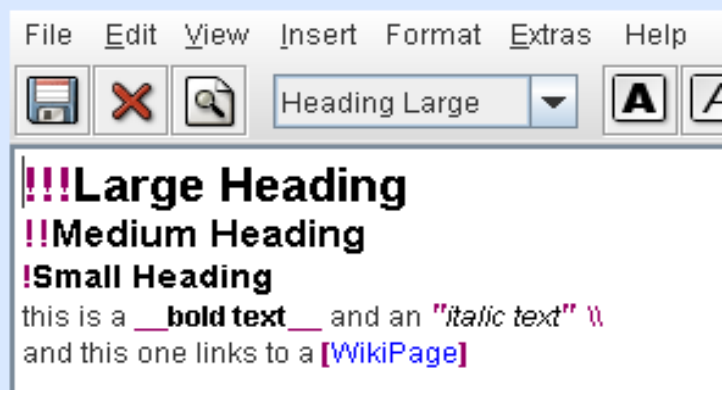

Image 1: Visual feedback of formatting in Wiki Wizard without hiding wiki markup

This approach was implemented by an editor called Wiki Wizard. WikiWizard is an open source editor for wikis under development at the i3G Institute of the Hochschule Heilbronn. It was released in a first stable version in March 2006. Currently it is only available for JSPWiki and can be integrated in the current beta release of this wiki engine. WikiWizard's design goal was to resemble something that the user already knows and allows him to get acquainted with it in an explorative manner without hiding wiki markup. So, we added toolbars and menus that users are familiar with when writing emails. JSPWiki's aproach to making wiki editing like writing email, providing the user with the ability to add attachments to every page comes as a help with this approach.

WikiWizard checks while the user is typing to see if the user is adding markup to the text and displays visual feedback immediately. For example, if you start a line with an exclamation point, the font instantly gets resized and the exclamation point get highlighted in red. If you add a second exclamation mark the font gets even bigger and so on. All markup characters are treated in this way and displayed in red so that you can easily distinguish markup from the actual text. At the same time, an outline in the sidebar displaying all headings is updated instantly. You can use this outline bar to jump to different sections and navigate in your document. WikiWizard also allows users to attach images, shows image previews and lets you insert those images into the text by double-clicking on them in the outline bar. It contains wizard dialogs for inserting links and emails and has a table editor for wiki tables. There's even a formatted paste option, that lets you copy formatted text, for example, from a Word document or webpage and paste it into the editor. The formatting then gets automatically translated into wiki markup.

WikiWizard is a proof of concept editor. There’s still a lot of room for improvement to catch up with the current state of the art WYSIWYG editors. For example there's no spell checker currently and preview images are currently only displayed in the sidebar, not in the text itself. Still it copes with the issues of a modern look and feel, exploitive usage and helping the user to keep the overview while editing long documents. What WikiWizard can not solve is the wiki markup mess.

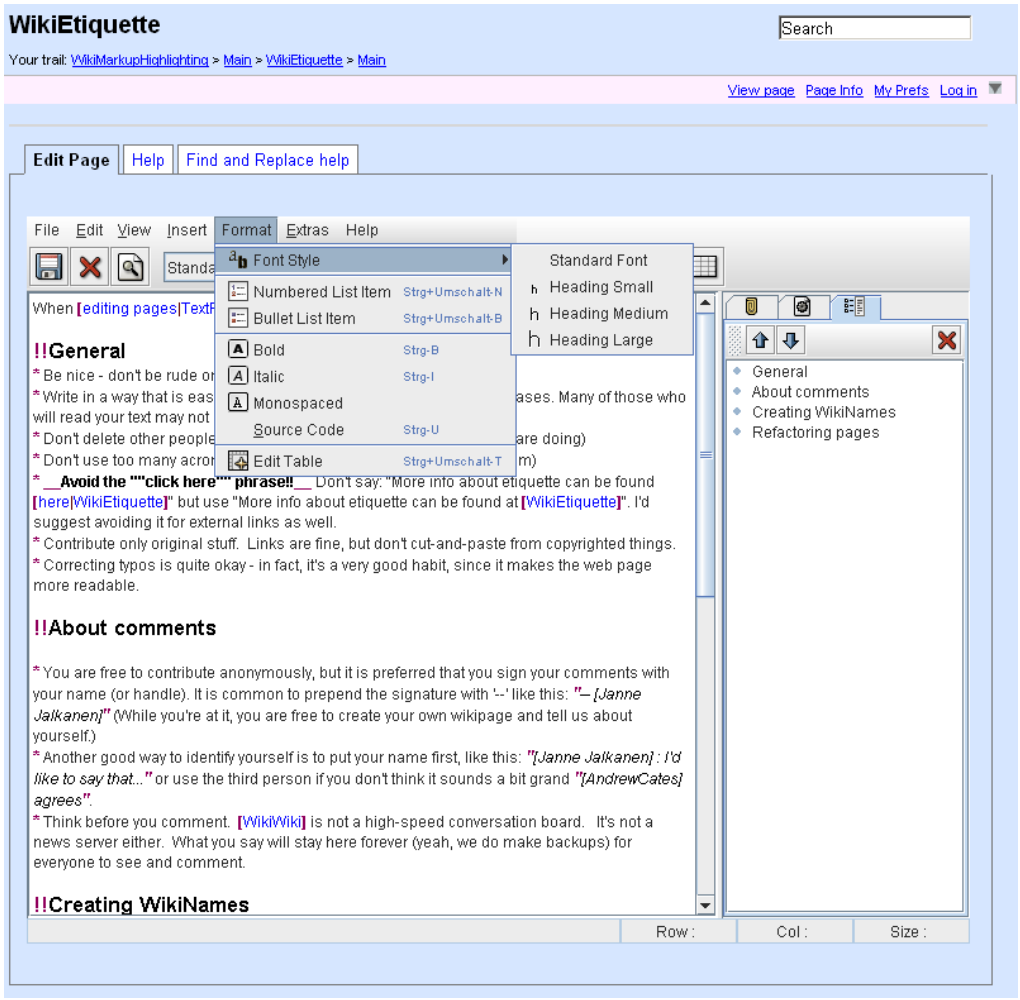

Image 2: Text editor with table of contents outline bar at the right side of the editor area

Conclusion

We have seen many obstacles that cause many to think that WYSIWYG is the only way to go. For now it also might be the only way to go for wiki engines that target enterprise markets where they have to compete with Microsoft's office suite and a user base that is used to working with it. But there should be an alternative to the visual approach for a new emerging market of knowledge workers who profit from the logical approach - What you see is what you mean. This WYSIWYM should not replace WYSIWYG but complement it in a different tier of the market. There should be editors that are in between the two extremes of plain old text editors and WYSIWYG only editors.

For this to work out there are still many hurdles to jump. First of all logical editing has to be brought back into public discussion. People have to know about the concepts before they can judge how to work with their documents. We need better tools to support editing in a modern way, where users don't miss any functionality except free formatting. Wiki parsers have to be more forgiving on syntax errors and wiki markup needs to be as intuitive and simple as possible. We have a wide variety of markup to choose from; we finally have to agree on the best elements of them all and create a common wiki markup standard, that is declarative like sql: say what you want, not how to do it. The markup available to do logical editing could not become much simpler, we just have to standardize it. We have to leave the door open for alternatives to visual editing.

If we close that door, we will miss a historic opportunity. We might fail to build the necessary infrastructure that will at one time let us move to a more productive document production, one that

uses the logical way, where what you see is what you mean. Without that, it will become much harder to advance wiki technology in general. The vision to write programs with easy scripting languages via "the wiki way" will become even harder. If we cannot teach users the logical approach of document production, how will it ever work out for programming for the masses? There should be a right for a computer literacy for everyone that is easy to learn and usable for anyone and independent of the technology platform. If we fail to achieve this, it is not the user's fault but ours, because we didn't deploy the simplest thing that could possibly work. If we succeed, we will see productivity advances in a field where we never would have expected them.

References

Lamport, Lesley, 24 Feb. 1987, *Document Production: Visual or Logical*, Notices of the American Mathematical Society (June 1987), 621-624.

Brooks Jr., F. P. 1987. *No silver bullet: Essence and accidents of software engineering*. Computer Magazine 20 (4): 10-19.

Cunningham, Ward, 19. Feb. 2004, *The Simplest way that it could possilby work*, Artima Developer

Désilets, Alain at al. 2005, *Are Wikis Usable,* Proceedings of the WikiSym 2005