T E C H N O T E :
On Creating Web-Friendly Documentation: A Few Techniques

By Tom Maremaa
tm@applelink.apple.com
Apple Developer Technical Support (DTS)

With the recent explosion of Web publishing, more and more documentation will be viewed primarily online, using your Netscape or Web browser. This signifies a rapid paradigm shift in both writing styles and content. Programmers and developers, in particular, need to understand that their forays into documentation must meet the test of online viewing, rather than hardcopy printing.

This Technote introduces a series of techniques designed to improve the quality of technical writing for a developer or programmer audience. It is aimed at content producers who must produce Release Notes, Readme files, and other forms of technical writing.

Integrating the Documentation Process Into Your Project

Writing documentation (docs) is usually something that comes at the end of a software or hardware project rather than at the beginning. Typically, it's done in the eleventh hour, like an afterthought rather than an integral part of the project. Imagine if you waited until the night before a code sample was due to start writing the code!

There are no formulas for producing documentation, particularly if it is aimed at a highly technical audience. A rule of thumb that I've found especially helpful in the creation of any documentation is to *allocate time for writing one page per day*.

If you factor one page per day into the equation for your project, you'll find yourself in less of a "pressure-cooker" situation as the release date nears. With this very simple step, and a commitment to follow through, you've overcome one of the largest stumbling blocks to creating lucid and comprehensible documentation.

Selecting Your Writing Tools

You want to match the tool to the project. To produce a first draft of any document, you need to use the tool that you're most comfortable with, i.e., a speedy word processor or text editor such as BBEdit. After producing the first draft comes the process of honing the document, a process which may require a more sophisticated tool, particularly if you are adding figures, tables, or footnotes.

1. Select your writing tools well in advance. You wouldn't think of programming without having a thorough understanding of your programming environment, including your compiler and debugging tools, would you? Show the same respect for your writing and editing tools.
2. Use a stylesheet.

Here is a very simple stylesheet created in ClarisWorks 4, which you can download.

This is a simplified version of the stylesheet we use to produce Macintosh Technotes, with three defined heading levels, a style for code snippets or samples, and bullet items. Adapt it to your project's needs.

You can be more productive, and be more likely to achieve your goal of writing one page per day, if you've already constructed the elements you need for a particular piece of documentation in a stylesheet.

3. Set a goal of producing 3 drafts — no more, no less — of your document before sending it out for review by your peers. Allow the first draft to be rough, the second for heavy-duty editing and fixing grammatical mistakes, and the third for polishing, spell-checking, and fine-tuning.
4. When you're satisfied that you have a draft that makes sense for your intended audience, save a version of it as an HTML file (which you can do very easily in ClarisWorks 4). Then, drag and drop that draft onto your Netscape 2.0 browser and view it locally.

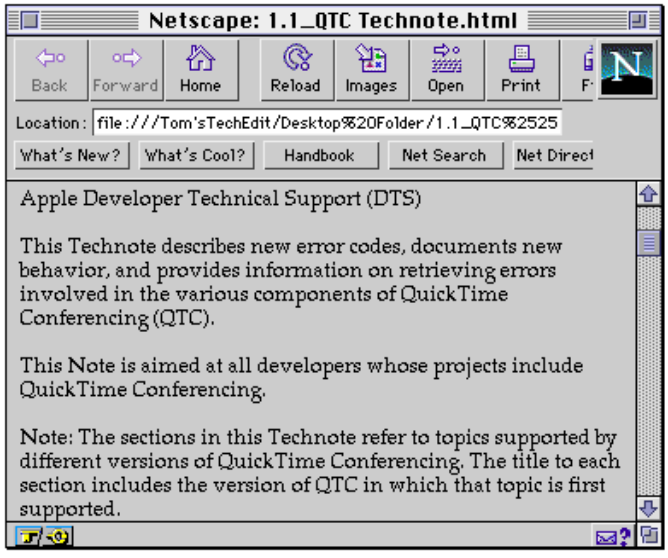
Consequences of Online Documentation

There are consequences to online documentation: 1) you must break down the information into smaller, often shorter sections, 2) you need to work with numbered lists, 3) your paragraphs need to be more concise, with a strong topic sentence which will catch the reader's eye and a concluding sentence that will resolve the issues or ideas raised in that paragraph, and lastly, 4) you'll have to work with a reader who has a shorter attention span.

Ultimately, as Web publishing becomes the standard, and you find your Netscape browser with a set of Web editing tools built in (such as you have with Adobe PageMill), your reader will also expect to find hypertext links embedded in your paragraphs. That's not just the wave of the future, it's today's reality.

If your document meets the HTML test, it'll have a definite outline form, with level headings, bulleted lists, and a clearly demarcated structure. Figure 1 shows an example of viewing a draft online.

Figure 1 An example of a draft viewed online



Working with a Specific Structure, an Example

It's important that your document follow a particular structure that the reader can anticipate while reading.

For example, Release Notes accompanying sample code, or an app you've built and want to share, may include the following sections:

- Introduction
- Building the Application
- Implementation
- Limitations & Known Problems
- Future Changes
- Summary
- References

Note that this structure is easy to follow, and includes navigational points for the reader.

Using Color Text for Heading Levels

If your document is going to be viewed onscreen, rather than printed, you can enhance its readability simply by adding blue and red heading levels, both to help the reader keep track of sections, and to prevent his or her being overwhelmed by a solid wall of text onscreen.

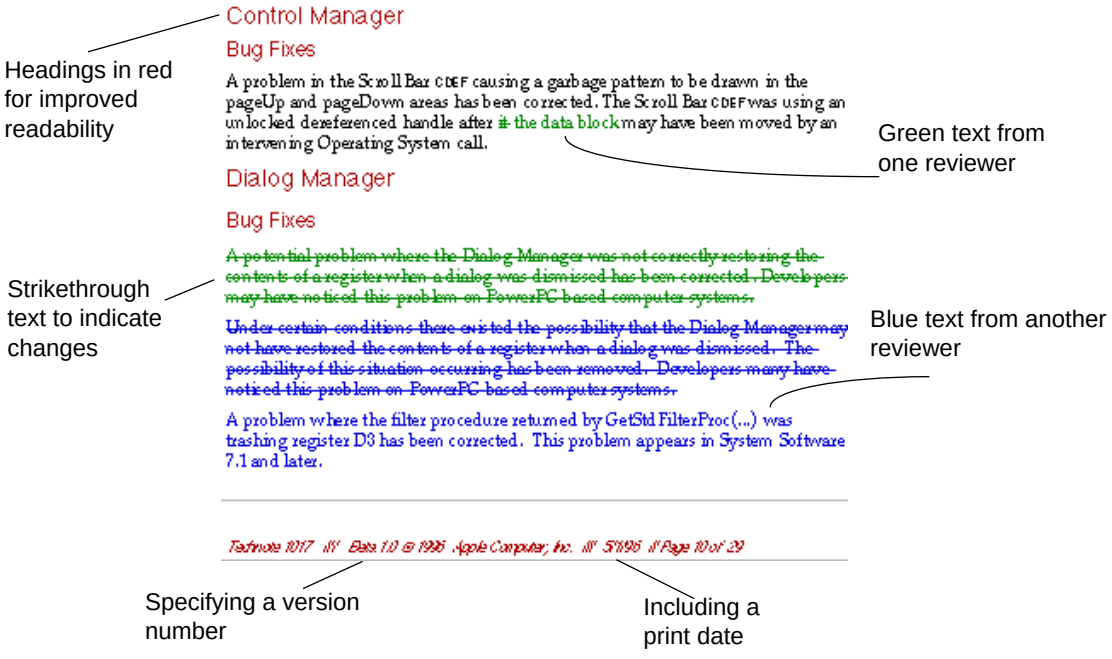
There are no rules that I know of for deciding which colors are preferable. Use your own judgment. For a long time, traditionalists disdained using color text in word processing documents, but I find that if my document is eventually to be converted to an Acrobat or another portable document file format, colored text goes a long way towards improving its readability. In fact, color text can even be a very useful method of getting reviewers to mark up a document, as explained in the next section.

Using Color Text in the Review Process

You can eliminate the hassle of printing 10 copies of your document, passing them around to reviewers, and getting back coffee-stained copies with indecipherable comments by using a different technique for review. Here are the steps.

1. Send out an electronic copy of your document in, for example, ClarisWorks 4, with the proviso that your reviewers mark up the copy electronically in a text color of their choice.
2. Specify that if they want to delete text, they use the strikethrough style, so that you can easily see what they recommend cutting. If they want to add material, specify that they use a text color to do so. Figure 2 shows an example of this.

Figure 2 Documentation sent out for review - an example

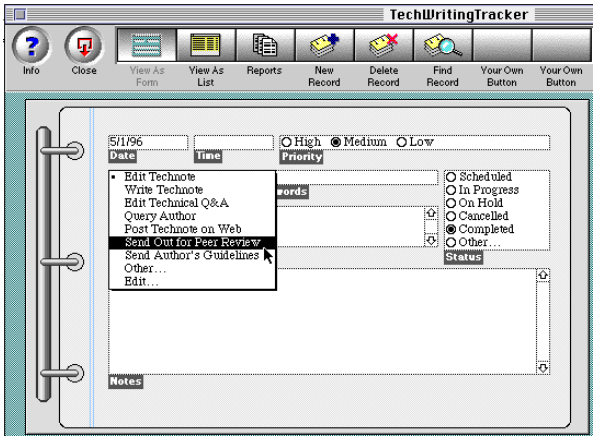


This technique results both in a speedier review process, and a more active involvement by the review team, which will improve the quality of the final document.

Tracking your Work

Just as you would keep a change log of code that you've written, it's important to keep a log of documents that you've produced, and their status, e.g., out for review, on hold, released, etc. Here is a little FileMakerPro 3 tracking database that I've created. The fields are modifiable, so you can create one to fit your own needs. Figure 3 shows a screen from the tracking database.

Figure 3 The TechWritingTracker



Download the FileMakerPro 3 TechWritingTracker by clicking on this icon.

Summary

The wave of the future for technical documentation is the Web, and it's already here. This change of venue has important consequences for writing and publishing. Documents that fail to meet the test of online readability will frustrate readers. The techniques described in this Technote ought to help you begin to write more Web-friendly documentation.

Further References

- Strunk, William, Jr., and E. B. White. *The Elements of Style*, third edition. New York: Macmillan, 1979. A short classic that offers practical advice on

achieving a clear and graceful expository style. One author calls this “a good book to read every year.”

Acknowledgments

Thanks to Frédérique Courard Hauri, Guillermo Ortiz, and Sunny Singha.