

Documenting the Mozilla Project: A Practical Example of Wikis in Open Source Documentation

Eric Shepherd
Mozilla Corporation
1981 Landings Drive, Bldg. K
Mountain View, CA 94043, USA
+1 650-903-0800
eshepherd@mozilla.com

ABSTRACT

This paper presents the practical experience of using wiki-based solutions for creating and managing developer documentation for the Mozilla project. It examines the documentation needs of the Mozilla project, then presents how the documentation project has evolved to its current MindTouch Deki based solution. In addition, practical ideas about the administration of an open source wiki-based documentation project are discussed.

Categories and Subject Descriptors

H.4.3 [Information Systems Applications: Communications applications]; H.5.3 [Information Interfaces and Presentation]: Group and organization interfaces — collaborative computing, Web-based interaction.

General Terms

Design, Documentation, Human Factors.

Keywords

Wikis, collaborative publishing, documentation, open source.

1. INTRODUCTION

The Mozilla project is a worldwide project, sponsored by the Mozilla Foundation [1], whose primary goal is to foster an open and participatory Internet through the development, use, and promotion of open standards. A key component of the project is the development of open source applications including the Firefox web browser and the Thunderbird email client.

In order to coordinate such a large open source software project, having readily-updated and thorough developer documentation is crucial. Orchestrating this massive documentation project is the purview of the Mozilla Developer Center [2]. The Mozilla Developer Center provides documentation on a wide array of development topics, including how to develop web sites using the latest web technologies, programming in JavaScript, and how to compile Firefox and contribute to the Mozilla project.

Other wikis at Mozilla are used for a wide variety of purposes, from product and architecture design and planning to the corporate intranet. However, this paper's focus is on the Mozilla Developer Center, and offers a look at the practical experience of managing a large documentation effort using wiki technology.

Section 2 examines the Mozilla project's documentation needs. Section 3 builds on this by discussing how Mozilla initially handled documentation before making the decision to transition to a wiki solution. In Section 4, we take a look at MindTouch Deki, our current solution for managing our documentation. Section 5

explores the positive and negative aspects to using a wiki for the documentation project. Section 6 provides an insight into the practical aspects of managing a large-scale open source documentation project using a wiki, and Section 7 offers some concluding thoughts.

2. THE NEEDS OF THE MOZILLA PROJECT

The Mozilla project attracts participants from around the world, with a wide array of skills and an equally varied amount of time available to contribute to the project. As such, it's important to consider the needs of the contributors when planning any work related to the project.

Similarly, it's important that the infrastructure behind the documentation be up to the job. In particular, the tools used to organize and maintain the content must be both capable of dealing with all contingencies and convenient for the documentation project's administrators to use.

2.1. Community Participation

Anyone in the Mozilla community must be able to contribute easily to the documentation. While many contributors to the documentation are highly technical people—including both developers and skilled technical writers—many other contributors are simply enthusiasts, who wish to contribute but may not have the technical background to write substantial amounts of new material. Instead, they offer their services by performing copy-editing and other tasks as suited to their abilities.

Encouraging this kind of participation is a key goal for the Mozilla Developer Center. By taking advantage of the writing skills of non-technical people, we can improve the quality of documentation contributed by highly technical people whose writing may not be clear.

To maximize the number of participants in the documentation project, it's important to offer a low barrier to entry by making the documentation easy to access and edit. However, it's necessary to balance this with a need for security so that inappropriate content does not get added to the documentation without being noticed and removed.

Another important aspect to the ability of the community to participate in the documentation project is communication. We make use of Internet Relay Chat (IRC) channels, mailing lists, and special discussion pages on the documentation site itself to share ideas, debate policies, and offer suggestions for new articles that need to be written.

Documentation “wish lists” are also maintained within the wiki itself, so that potential contributors can easily find suggestions for articles that they might be able to write and contribute.

The Mozilla Developer Center web site has over 40,000 registered users (which doesn't include people that read the site anonymously), and receives over 650,000 page views per day. The software back-end needs to be able to respond effectively to this amount of traffic.

2.2. Ease of Maintenance

Due to the large scale of the documentation and the wide variety of topics covered, it's important that maintaining the documentation be as easy as possible.

With anywhere from dozens to hundreds of contributed edits performed per day, being able to quickly and easily verify the changes while performing the other maintenance tasks that are necessary is crucial. This is especially important if the people managing the project are also contributing documentation; they have other tasks that need to be done in addition to administrating the content.

2.3. Security

As mentioned briefly above, it's important to balance a need for security with the need to offer easy access to readers and contributors alike.

There are a number of different security issues that need to be considered.

2.3.1. “Spam”

Spam is a constant source of frustration for wiki administrators. Between people who thoughtlessly post irrelevant or inappropriate content to the documentation site and automated “bots” that attack the site and post commercial notices, it's important to be able to quickly and easily identify and remove this type of edit.

2.3.2. Sabotage

An unfortunate reality is that any publicly-editable document on the Internet is prone to attract unscrupulous people who will cause damage to the content on purpose, either due to a misguided sense of humor or because of an agenda of some nature. This sort of problem can be tricky to spot if the sabotage is subtle. Fortunately, most sabotage involves massive and very obvious changes that are easy to revert.

As an example, in 2007 the Mozilla Developer Center experienced a wave of automated attacks in which “bots” would create new accounts and then proceed to access pages at random, deleting sections from them.

2.3.3. User Information Security

Any web site that requires log-in information from its users may obtain personal information that must be kept secure. In the case of the Mozilla Developer Center, this is a minimal set of information, consisting only of the user's name, password, and email address.

3. A BRIEF HISTORY OF MOZILLA DOCUMENTATION

Before switching to a wiki-based solution, Mozilla's developer documentation was maintained in HTML and version controlled using the Concurrent Versions System (CVS).

The content was editable on the web using Doctor, a utility that allowed the user to edit articles' HTML in the browser, or could be checked out using CVS and edited using the writer's preferred text editor.

However, the result was a high barrier to entry, as it required an understanding of HTML to contribute fully to the documentation project, and there was a certain degree of opacity to the documentation, in that there were not readily visible logs of changes unless you used CVS.

The result was a relatively stagnant documentation project, with contributions primarily offered by the software engineers and rarely updated after the initial writing. It became clear that this was a serious problem, and it was decided that a wiki would make a better solution for Mozilla's developer documentation.

MediaWiki, the software that powers the popular Wikipedia online encyclopedia, was selected to drive the new Mozilla Developer Center web site due to its popularity and maturity at the time.

In 2005, the Mozilla Developer Center was launched, and rapidly grew. By 2007, it was getting well over 500,000 page views per day and contained over 13,000 articles in thirteen languages.



Picture 1: The MediaWiki-powered Mozilla Developer Center.

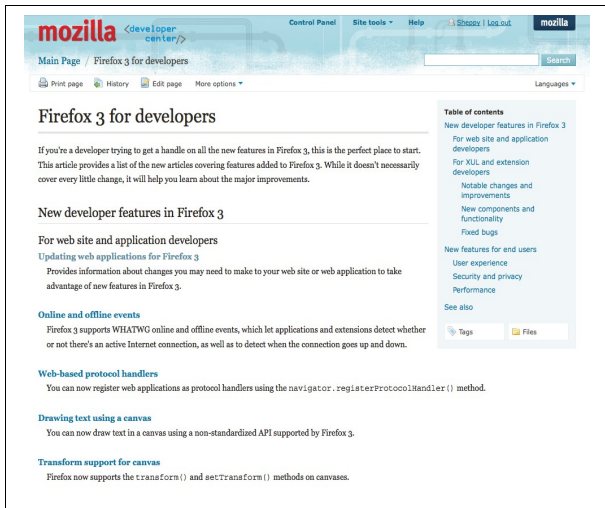
Over time, it became apparent that there were some drawbacks to using MediaWiki for the Mozilla Developer Center. Key among these was the need to patch each version with a series of minor changes to customize the wiki's behavior to better suit our needs.

In addition, MediaWiki did not provide a good way to handle downloadable code samples. While it was possible, in theory, to upload them using the image uploading feature, this turned out not to work well in practice as it was confusing for our users, and tended to lend itself to abuse. Instead, code samples had to be emailed to an administrator, who would upload them to the server using a Secure File Transfer Protocol (SFTP) client.

4. MINDTOUCH DEKI ADOPTION

By early in 2007, it became clear that MediaWiki was no longer a viable long term solution, so a search began to select a new wiki engine to power our documentation. After some research, the decision was made to migrate the Mozilla Developer Center to MindTouch Deki [3]. This section examines MindTouch Deki in

brief, as well as the reasons why Mozilla selected it to power its documentation going forward.



Picture 2: The MindTouch Deki powered MDC.

4.1. Open Source

As a strong proponent of open source, one of Mozilla's top priorities was selecting a documentation management solution that was open source. Both MediaWiki and MindTouch Deki are licensed under the GNU General Public License Agreement (GPL).

Mozilla is using the Enterprise edition of MindTouch Deki. The only difference between the Enterprise version and the free version is that the Enterprise version includes support from MindTouch itself.

4.2. Ease of Maintenance

MindTouch Deki offers a very easy to use control panel for administrating the wiki. While it does not offer quite as expansive a selection of features as MediaWiki, the available features are more than adequate for Mozilla's needs, and are improving rapidly with each release of the software.

MindTouch Deki also offers Really Simple Syndication (RSS) feeds, whereby editors can monitor for all changes to the documentation, changes to specific pages of interest by adding those pages to a watch list, or even changes contributed by specific users.

This is actually one of the weaker areas in MindTouch Deki right now; the tools for comparing versions of documents are not as mature as those in MediaWiki; in particular, changes are listed inline instead of side-by-side, which makes certain types of editing more complicated. Additionally, change logs show entire documents, even when only one word has changed, which makes reviewing changes more time-consuming than necessary.

We are working with MindTouch to improve these features, and hope that their next major release will address our concerns here.

4.3. Ease of Use

There are several usability gains in switching to MindTouch Deki. Foremost among these is a greatly reduced barrier to entry. Unlike MediaWiki, which requires the user to learn to use

“wikitext”—a simple layout language—in order to format the content, MindTouch Deki offers a what you see is what you get (WYSIWYG) editor that makes it very easy for new contributors to feel at home.

For more advanced contributors, MindTouch Deki allows the writer to edit in source mode, and the source is in XHTML format, which many—perhaps even most—Mozilla Developer Center contributors are familiar with. This flexibility was a key advantage in favor of MindTouch Deki over MediaWiki.

Another advantage to MindTouch Deki is its excellent support for attaching files to articles; this makes it easy to distribute relevant sample code and demonstrations, keeping them within the context of the topic at hand.

Also, MindTouch Deki has integrated support for source code syntax highlighting. This allows our contributors to embed snippets of source code into articles and enable syntax highlighting—including line numbers—to make the code both easier to read and easier to reference from the body of the article. Under MediaWiki, code snippets were simply contained in pre-formatted text blocks.



Picture 3: Syntax highlighted JavaScript code in an article.

Another advantage to MindTouch Deki, for the Mozilla Developer Center, is the fact that a single wiki can support multiple languages. When we used MediaWiki, it was necessary to have a separate wiki for each language (which resulted in the need to patch the software to support cross-linking between localizations). MindTouch integrates multiple language support quite effectively, allowing users to search in either their preferred language or any other, falling back to English if they fail to find a result in their native language.



Picture 4: Editing an article in Japanese.

There are currently some gaps in the multi-language support; in particular, there is not yet support for RSS feeds of changes only in specific languages; instead, the feeds include content for all localizations, resulting in information overload, since most users are only interested in changes in one or two languages at most. However, MindTouch is aware of these issues and should be addressing them soon.

4.4. Security

MindTouch Deki offers several features that help maintain the security and viability of our documentation. Automated attacks are prevented through the use of captchas, which make it much more difficult for bots to infiltrate the system.

The software offers the ability to easily find and restore deleted pages and files, to undo edits, and provides the ability to ban users both by username and IP address, so that attacks can not only be spotted and corrected, but the perpetrators punished and prohibited from repeating their offenses.

This is not an area in which MediaWiki was deficient. In fact, MindTouch had to add the username and IP banning support for us in order to offer the same capabilities as MediaWiki.

4.5. Ease of Customization

In the past, it had been necessary to patch MediaWiki, in addition to installing extensions, in order to make the software approach fulfilling all of our needs.

Unfortunately, this led to lengthy delays in updating the software, as our patches were frequently incompatible with updated versions of the MediaWiki software, and several extensions we used were not updated to be compatible with new releases of MediaWiki on a regular basis.

MindTouch Deki is more easily customizable. Although there aren't nearly as many extant extensions for the platform, it will be easier going forward for us to create our own.

One particularly useful area of customization is the editor's toolbar. Unlike in MediaWiki, it's trivial to customize the editor toolbar in MindTouch Deki. We simply use the wiki's control panel to edit a toolbar customization file that allows us to change the set of styles, fonts, and so forth offered by the toolbar.

By customizing the toolbar, we can make it possible for the user to easily conform to our layout and style standards. The toolbar offers style options for note and warning callouts, variable and function names, and so forth.

In addition, MindTouch Deki provides support for web services, which allows customization of the content by utilizing capabilities offered by other web sites, such as Digg, Flickr, and Windows Live. We expect to take advantage of this ability going forward, and hope to, for example, integrate the Bugzilla [4] bug database software used by Mozilla with our documentation. This will make it easier for documentation to make reference to bugs or sets of bugs that are relevant to the topic being covered.

4.6. Support

From the outset, MindTouch was very enthusiastic about working with Mozilla. They agreed to make a number of improvements to their software in order to add features needed by the Mozilla Developer Center, thereby allowing us to focus on the actual documentation itself.

Once we committed to switching to MindTouch Deki, we signed a support contract with them, which was well worth the money. From the beginning, MindTouch has been extremely responsive with support and has made innumerable changes to their software, from simple bug fixes to adding substantial new features.

In addition, MindTouch created a utility to convert MediaWiki sites into MindTouch Deki format, and did the conversion work for us.

Having this sort of enthusiastic support took the burden off our small team, which was an enormous help during the transition from MediaWiki to MindTouch Deki.

Our hope is that through Mozilla's backing, MindTouch Deki can become a popular solution for open source projects. By using our resources to fund and guide the development of the software, we can help MindTouch reach a wider audience in the open source community. The changes we request—and receive—are generally useful for any open source documentation project, and are available to anyone that adopts MindTouch Deki.

5. ADVANTAGES AND DISADVANTAGES

There are both positive and negative sides to using a wiki to manage a large documentation project such as the Mozilla Developer Center. This section examines both sides of the issue.

5.1. Advantages

Because anyone that reads the documentation may contribute, we are able to encourage developers, users, and casual readers alike to contribute as their capabilities allow. Developers are encouraged to write articles covering topics with which they are familiar, but if they prefer to let others do the initial writing, they can easily go through and correct factual errors once the document is created.

Similarly, casual readers may not have the technical background to identify and correct factual errors (especially subtle ones), but they frequently notice and fix spelling and grammatical errors. Equally as importantly, readers often find gaps in the documentation, where material that's needed in order to fully understand a topic is missing. This enables the technical contributors, whose knowledge may be extensive enough that they don't notice the missing content, to create or solicit submissions to fill the gaps.

Another key advantage to using a wiki for documentation is the ability to more easily offer translations of content. Volunteers produce all of our translated content, and Mozilla Developer Center documentation is offered in nine languages, although to varying degrees of thoroughness. Our localization teams for Japanese, Polish, and Spanish are particularly active.

This sort of easy access for contributors lets Mozilla's documentation be as open as the rest of the project.

5.2. Disadvantages

As mentioned previously, sabotage and spam are frequent problems on publicly-editable web sites, including wikis, and it's necessary to watch closely for occurrences of these.

Along a similar vein, occasionally people who don't understand a topic fully will contribute inaccurate content. They don't mean any harm, yet can have lasting repercussions on the quality of the documentation if their errors are not spotted and corrected quickly.

Another problem that can arise—although fortunately not often in the case of the Mozilla Developer Center—is that people with different styles and agendas can wind up in conflicts that negatively impact the quality of the content or the coherency of the community behind the documentation effort.

There are several ways this particular problem can arise. Disputes can occur when two or more contributors disagree about the necessity of a certain piece of content, for example. This can either result in a lively debate in the discussion area of the wiki or a “war” in which two contributors repeatedly edit and re-edit the same material, each trying to outdo the other.

It's also possible for people with widely different writing styles to work on the same document, resulting in occasional confusion.

The next section will look at how the Mozilla Developer Center is administered to help minimize the impact of these disadvantages, while taking the maximum possible advantages of all the wiki has to offer.

6. MANAGING THE MOZILLA DEVELOPER CENTER

Managing the Mozilla Developer Center—and, presumably, any large-scale documentation project—involves a great deal of knowing when not to do anything. It's important to provide guidance at all times, gently steering the documentation project in a sane direction, without stifling the creativity and enthusiasm of the community.

6.1. Organization and Guidance

The leader of a wiki-based open documentation project needs to provide guidance to its contributors. This starts with the basics: creating a set of guidelines that should be followed when creating documentation.

In the case of the Mozilla Developer Center, we have an assortment of rules and guidelines to help ensure the quality of our documentation. This also helps make the documentation as internally consistent as possible, which improves the reader's experience while searching and reading the content.

6.1.1. Grammar and Spelling Rules

The Mozilla Developer Center's English edition, along with each localized edition, lays out specific guidelines for the grammar and spelling rules authors are asked to follow [5]. For example, we

recommend specific style guides and dictionaries so authors can all have a standard set of resources from which to draw conclusions as to questions of linguistic style.

6.1.2. Style and Formatting Rules

In addition, we offer recommendations for how to handle abbreviations, the fact that we welcome the use of contractions, and so forth.

Guidance is also offered in terms of how to format and lay out context for ease of readability and conformance with our overall look and feel.

6.1.3. Content Organization and Hierarchy

In order to keep the documentation's organization in check, and to avoid content being “lost” among the sea of articles on the Mozilla Developer Center site, a tree of pages offering tables of contents of subsections of the site is provided. Starting at the home page of the wiki site, each click leads to another table listing articles covering increasingly specific topics.

This serves multiple purposes. It allows readers who do not necessarily care for searching—or who don't know exactly how to search for what they need—to track down information. It also lets writers find more quickly what content does not yet exist, as well as to link new content into the hierarchy to make it easier for others to find.

6.1.4. Examples and Templates

The Mozilla Developer Center offers an assortment of MindTouch Deki templates—which are something like macros—that can be used to automatically format various types of material, such as special notes, code samples, and the like in a standard manner. This helps writers more quickly generate content that fits our style guidelines.

Additionally, certain types of articles that follow a standard format and layout often have prototypes that can be copied, pasted, and edited to suit the writer's needs. For example, when creating an article covering a new interface, authors are encouraged to base it upon a sample page [6], which is accompanied by an article describing its contents and how to customize it [7].

This helps to ensure consistency among newly created documentation, and an effort is underway to go back and update documentation written before the standard format was devised to match the new style.

6.2. Monitoring the Content

One of the most time-consuming parts of the job when managing a wiki-based documentation project is that of monitoring the content to ensure its accuracy and legitimacy. There are several aspects of this job, including proofreading, cleanup work, constant oversight of the organization of the content, and dealing with troublemakers.

Fortunately, wikis include a wide variety of tools to help monitor content, such as the ability to watch pages for changes, receiving email when given pages are changed, or RSS feeds that inform interested parties when pages change. Tools are also provided that allow editors to compare two versions of an article to identify exactly what changed, and to revert incorrect or inappropriate changes.

6.2.1. Proofreading and Clean-up

Because people with various levels of proficiency at writing contribute to the documentation, it's important to proofread articles as they come in. Over time, the editors learn which people tend to contribute articles that only need a very brief look, and which people's work needs a heavy hand to ensure its legibility.

Even writers with a great deal of talent will often contribute articles that don't adhere to the style guidelines. In these cases, it becomes necessary to retouch the material to correct formatting and layout issues.

For example, some contributors will write their article in an offline text editor, then simply copy and paste the material into the wiki without doing any formatting work at all, in which case it may wind up being up to the editor to go through the piece and apply the appropriate formats to headers, to style code samples correctly, and so forth.

6.2.2. Keeping the Organization Intact

With a large number of contributors, it's easy for the content to become disorganized. There are times when people look for an article about a given topic, fail for whatever reason to find it, and contribute a new article covering the same information.

In other cases, new articles are submitted without properly tagging them with appropriate categories, or without inserting them into the correct table or tables of contents.

It's therefore important for the documentation's editors to watch for newly created articles and ensure that they're necessary, and if they are, to ensure that they're properly categorized and indexed so that readers can find them.

It's also sometimes necessary to determine which of two (or sometimes more) articles covering the same topic is the best, and to either delete the extra article or merge useful pieces of its content into the other.

6.2.3. Dealing With Troublemakers

Unfortunately, troublemakers do crop up from time to time. It's necessary to have a policy in place for dealing with them.

There are several categories of troublemakers. The easiest to deal with are those that insert spam or other irrelevant material into the documentation. Using the tools provided with the software, it's easy to spot changes that are made to articles, and spam usually stands out very clearly as you review the list of changes. Once identified, it's typically easy to roll back the edit and ban the offending user from editing the site again.

More difficult to deal with are arguments among contributors. These have fortunately been rare in my experience working with the Mozilla community, but they do arise. Typically, I choose to handle these by allowing the debate to go on until I sense that tensions are starting to rise, then I step in and make a decision based on what I've seen, trusting the community to follow my lead.

This has generally worked very well; even the people who argued against whatever decision I wind up taking have usually had respect for my willingness to listen. I believe that the fact that I stay out of the debates until the end helps ensure my credibility as an unbiased decision-maker when I do finally become involved.

When that approach fails, it's often necessary to talk to the involved parties to help them accept the decision and to assure them that there is nothing personal about the final decision.

6.3. Cheerleading

Another important job for the administrators of an open documentation project is that of cheerleader: it's important to encourage the contributors to keep working, and to offer helpful advice and suggestions when it's called for.

This can be related to the job of editing, in that sometimes hurt feelings need to be addressed when someone loses a debate about an issue related to the documentation, or when someone's article gets corrected by someone else; some people don't like their work being edited by others.

Another form of cheerleading involves encouraging writers to stick to the style guidelines. This needs to be done gently, to avoid discouraging future participation in the project.

7. CONCLUSION

Managing an open documentation project requires diplomacy and tact, as well as a strong knowledge of the project, of writing, and of the community participating in the project. Keeping the feelings of the contributors in mind at all times goes a long way toward ensuring success.

By taking full advantage both of the capabilities of the wiki behind your documentation as well as of the capabilities of the community involved in the project, a wiki can be a powerful tool to create exceptionally useful documentation.

REFERENCES

- [1] *The Mozilla Foundation*. Accessed last: 2008.
<http://www.mozilla.org/foundation/>.
- [2] *Mozilla Developer Center*. Accessed last: 2008.
<http://developer.mozilla.org/>.
- [3] *MindTouch Deki*. Accessed last: 2008.
<http://wiki.mindtouch.com/>.
- [4] *Bugzilla*. Accessed last: 2008.
<http://www.bugzilla.org/>.
- [5] *MDC: Writer's Guide*. Accessed last: 2008.
http://developer.mozilla.org/Project:En/Writer's_guide.
- [6] *MDC: Sample interface document*. Accessed last: 2008.
http://developer.mozilla.org/Project:En/Sample_interface_document.
- [7] *MDC: Writing interface reference documentation*. Accessed last: 2008.
http://developer.mozilla.org/Project:En/Writing_interface_reference_documentation.