

Zend Wiki Manual - No component - Andries Seutens

<ac:macro ac:name="unmigrated-inline-wiki-markup"><ac:plain-text-body><![CDATA[

<ac:macro ac:name="unmigrated-inline-wiki-markup"><ac:plain-text-body><![CDATA[

Zend Framework: Zend Wiki Manual - No component Component Proposal

Proposed Component Name	Zend Wiki Manual - No component
Developer Notes	http://framework.zend.com/wiki/display/ZFDEV/Zend Wiki Manual - No component
Proposers	Andries Seutens Gavin (Zend-liaison)
Revision	6 October 2006: Updated from community comments. (wiki revision: 22)

Table of Contents

1. Overview
2. References
3. Component Requirements, Constraints, and Acceptance Criteria
4. Dependencies on Other Framework Components
5. Theory of Operation
6. Milestones / Tasks
7. Class Index
8. Use Cases
9. Class Skeletons

1. Overview

We have documented numerous cases and examples where the current process imposes too many hurdles, complications, and effort for those wishing to use it and contribute. Since the vendor of our current wiki successfully uses their own product for their manual, including generating the PDF and HTML, we would like to consider the implications of switching from authoring content in docbook to using our wiki.

We are proposing opening up the English documentation process to participation by all documentation team members, and allowing wiki page commenting by anyone, but with reasonable moderation by the documentation team members, following in the footsteps of wikipedia. In order to help achieve consistency, we are also proposing borrowing mozilla.org's documentation style guide, subject to exceptions you might recommend (hint).

The process for adding content and editing the ZF manual pages might benefit from the use of a wiki that allows user-contributed comments. How many times have you wanted to make a suggestion or change to the English manual, but not done so?

2. References

- [Wikification of ZF Manual Project Page](#)
- [Initial test import for the Zend Framework Manual](#)
- [Confluence for professional documentation](#)
- [Drupal](#)

3. Component Requirements, Constraints, and Acceptance Criteria

- Reduce the learning curve and effort required to make edits to the manual and translations.
- Allow anyone to comment on each section, but with comment moderation, so that the manual can evolve more quickly and with more flexibility than the current process.
- Follow a documentation style guide and proofread before it is released.

4. Dependencies on Other Framework Components

- For the initial import docbook files are required

5. Theory of Operation

The component is instantiated with a mind-link that ...

6. Milestones / Tasks

For the mechanics of this new process, we setup two new wikis. One would contain the last official release for public viewing and commenting. The other would be a draft, work-in-progress with comments rolled into the body of the page before pages are "released". When we prepare for a release, changed pages could be copied to the official release wiki. Then, translators could use the wiki "difference" feature to see what has changed in the English manual on the release wiki, without seeing all the interim edits made in the draft wiki.

There might be a wiki plugin that allows tagging and viewing differences between tags (<http://confluence.atlassian.com/display/CONFEXT/Home>). If so, such a feature/plugin might eliminate the need for having two wikis. I am not eager to consider using a different wiki, as it would significantly complicate user account administration and logins. The current wiki already supports UTF8 text.

Our current wiki system supports export to PDF and HTML, although the exports are "plain", and might require some creative manipulation, such as using Zend_Pdf to beautify it. I agree entirely with recent posts about the value of providing HTML and PDF downloads for the manual. Several questions remain, including the docbook import/export process for our wiki (Confluence).

Also, incubator components using the wiki for documentation would make documentation available immediately, without requiring the curious to go through the process of "building the manual". Many of us would like to help with documentation for the new incubator components. Group editing of a wiki page has advantages over trying to coordinate editing of the documentation via SVN.

Milestones

- Quick access: type http://framework.zend.com/zend_controller similar to php.net, where you can easily access a specific function by appending it to the url
- Unreleased incubator documents should be available in wiki, but not added when extracting to HTML and PDF (Maybe tagging or a separated wiki space).
- Some contributors are writing like "this function can handle xxxx" other as "you can handle xxxx with this function"... note the first person style... a defined write style is must and makes the document flow smoothly as a whole.
- Automatically generate HTML and PDF for each language, either directly from the wiki, or from docbook, after auto-generating docbook from the wiki.
- Improve syntax highlighting

Tasks

- Study <http://confluence.atlassian.com/x/8hIC> information on features, including the user manual
- Research [Confluence for professional documentation](#)

- Research plugins and current efforts to export the wiki to docbook. <http://confluence.atlassian.com/display/CONFEXT/Home>
- Research possibility of tagging pages, such that a PDF could be created from tagged pages.
- Consider adding "edit last comment" to our issue tracker (Jira), and search for a way to do so with our wiki (Confluence).
- Decide on moderation process / policy for comments.
- Provide feedback to an example docbook to wiki converted page: <http://framework.zend.com/wiki/x/9hE>
- Provide feedback to an example new wiki homepage: <http://framework.zend.com/wiki/x/ahl>

Process Mechanics

The purpose of the wikification includes removing the need for any of us to write docbook.

1. Docbook to wiki import (one time only)
2. Translators and english manual authors use the wiki to author content, but they should restrict and limit themselves to the basic wiki markup already found in the wikified manual pages.
3. All CLA signers use the wiki to comment
4. Translators and english manual authors review comments, sometimes deleting comments and merging comments with the wiki page content
5. Wiki to docbook export (every time we make a release of the ZF)
6. We use normal docbook to HTML/PDF process to publish updates to <http://framework.zend.com/manual>
7. If we can export the wiki to HTML/PDF and they look as nice as #6 above, then we can use the HTML/PDF created by the wiki to HTML/PDF process. The wiki to HTML process could be either something custom, using Zend_Pdf, or using Confluence export. However, we still don't want to lose the ability to convert back to docbook, just in case we need the docbook formatted version of the manual for some other reason. To keep this ability and make wiki to docbook conversion easier, wiki authors must not use "fancy" wiki markup (use only basic tags that are already used in wikified manual).

7. Class Index

No classes are required for this proposal

8. Use Cases

No use cases are required for this proposal

9. Class Skeletons

No class skeletons are required for this proposal

```
]]></ac:plain-text-body></ac:macro>
]]></ac:plain-text-body></ac:macro>
```