Zend_Filter_Bbcode - Pieter Kokx

<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_Filter_Bbcode Component Proposal

Proposed Component Name	Zend_Filter_Bbcode
Developer Notes	http://framework.zend.com/wiki/display/ZFDEV/Zend_Filter_Bbcode
Proposers	Pieter Kokx
Revision	1.1 - 1 August 2006: Updated from community comments. 1.2 - 27 December 2007: Started to update the proposal 1.3 - 29 December 2007: Updated the class skeleton. (wiki revision: 34)

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

Zend_Filter_Bbcode is a stack-based bbcode parser which will always output valid XHTML. This component will also implement context-awareness for each tag.

2. References

- BBCode Wikipedia

3. Component Requirements, Constraints, and Acceptance Criteria

- This component **will** provide extensibility to create your own BBcode tags.
- This component **will** provide XHTML valid output.
- This component **will** provide context-awareness for BBcode tags.

4. Dependencies on Other Framework Components

- Zend_Filter_Exception
- Zend_Filter_Interface
- Zend_Filter
- Zend_Loader
- Zend_Uri
- Zend_View_Helper_*

5. Theory of Operation

Many programmers are parsing BBcode with regular expressions. Well, regular expressions isn't the worst solution for this problem, but it is almost an impossible task to create XHTML valid output for user input like '[b][u]some sample test[/b][/u]'. A well written stack-based BBcode parser (like this proposal) does produce XHTML valid output in that case.

With regular expressions, extensibility is also a huge problem. When you are looking back to regular expressions you've written a month ago, mostly you will keep staring at it for half an hour and still not understand it. With Zend_Filter_Bbcode, you just have to extend Zend_Filter_Bbcode and add a function (see UC 3). You can even use PHP code for the tag, so it isn't a problem to add line numbers or highlighting to a tag like [code].

Context-awareness is also a main concern. HTML doesn't allow text inside a <ol> or <ul> tag, unless it is inside a <li> tag. When you are using regular expressions or just simple replaces for a BBcode parser, this is an impossible task. But Zend_Filter_Bbcode checks the nesting of a tag, and also if it is allowed in the current position. If it isn't allowed, the tag will not be parsed. The same is with text in the wrong position, if it isn't allowed there, it will not be added to the return string.

6. Milestones / Tasks

- Milestone 1: [DONE] Initial class design
- Milestone 2: [DONE] Create prototype
- Milestone 3: Finish proposal and submit for community review
- Milestone 3: Create code-covering unit tests

7. Class Index

- Zend_Filter_Bbcode
- Zend_View_Helper_Bbcode

8. Use Cases

9. Class Skeletons

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