Zend_Commons - Yegor Bugayenko

<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_Commons Component Proposal

Proposed Component Name	Zend_Commons
Developer Notes	http://framework.zend.com/wiki/display/ZFDEV/Zend_Commons
Proposers	Yegor Bugayenko
Zend Liaison	TBD
Revision	0.1 - 28 February 2011: Initial Draft. (wiki revision: 19)

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_Commons is a collection of object-oriented strategies wrapping most common algorithms and operations. The biggest problem with existing PHP functions like `fopen()`, `file_get_contents()`, `shell_exec()`, etc. is that they are not object-oriented and are a huge legacy from PHP4. Zend_Commons resolves this problem by means of an extendable collection of classes designed in a Strategy pattern.

Moreover, this approach is very helpful during unit testing, when sensitive I/O, system, HTTP, etc. operations shall be properly isolated and mocked. When every operation is presented as an object retrieved from a factory - we can control their behavior to the very last detail.

I can't imagine right now all possible algorithms we may wrap in Zend_Commons. The idea is to create a factory of reusable components, which can be extended later by other contributors.

2. References

- apache commons

3. Component Requirements, Constraints, and Acceptance Criteria

- This component **will** implement algorithms and operations.
- This component **will** include a unified entry point.
- This component **will** allow customization.

4. Dependencies on Other Framework Components

- Zend_Exception
- Zend_Log
- Zend_Loader_PluginLoader

5. Theory of Operation

One of the strategies is instantiated via factory method, configured, and executed. Every strategy is a plugin, loaded with `Zend_Loader_Plugin`
.

6. Milestones / Tasks

- Milestone 1: Design notes will be published
- Milestone 2: Working prototype checked into the incubator
- Milestone 3: Unit tests exist, work, and are checked into SVN.
- Milestone 4: Initial documentation exists.

7. Class Index

- Zend_Commons
    - Zend_Commons_Exception
    - Zend_Commons_Abstract
    - Zend_Commons_Listener
    - Zend_Commons_Event
        - Zend_Commons_Io_Read
        - Zend_Commons_Io_Write
        - Zend_Commons_Io_Delete
        - Zend_Commons_Io_Mkdir
        - Zend_Commons_Io_Chmod
    - Zend_Commons_Exec
    - Zend_Commons_Email
    - Zend_Commons_Http
- more later...

8. Use Cases

9. Class Skeletons

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