Zend_File_Convert

<ac:macro ac:name="note"><ac:parameter ac:name="title">Under Construction</ac:parameter><ac:rich-text-body>
This proposal is under construction and is not ready for review. </ac:rich-text-body></ac:macro>

<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_File_Convert Component Proposal

Proposed Component Name	Zend_File_Convert
Developer Notes	http://framework.zend.com/wiki/display/ZFDEV/Zend_File_Convert
Proposers	Jonathan Bond-Caron
Revision	1.1 - 1 August 2006: Updated from community comments. (wiki revision: 5)

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_File_Convert is a library that aims to make conversion between file formats simple and efficient. As an example, the initial focus of this proposal will be on an ambitious conversion of XHTML 1.0 (http://www.w3.org/TR/xhtml1/) documents to PDF.

2. References

- DomPDF

3. Component Requirements, Constraints, and Acceptance Criteria

The first step will consist of creating a factory class that tries to convert file A to file B. Following are list of requirements:

will include a factory method.
will not allow subclassing.
will allow adding file converters
will be able to do automatic detection of the file mime-type

will throw an exception if no valid file converter is found
will not use a cache, this is job of each Zend_File_Converter

4. Dependencies on Other Framework Components

- Zend_Exception
- Zend_File ?

5. Theory of Operation

The Zend file converter will try to find the appropriate converter (Zend_File_Converter) to make the conversion between two files. If will then call the appropriate converter to make the appropriate file conversion. If no valid converter is found, the class will throw an error.

6. Milestones / Tasks

- Milestone 1: Publish proposal.
- Milestone 2: Revise proposal, approve for Incubator development.
- Milestone 3: Commit working prototype to Incubator.
- Milestone 4: Commit working unit tests.
- Milestone 5: Write up a simple Zend_File_Converter implementation to use as an example.
- Milestone 6: Write end-user documentation.
- Milestone 7: Revise implementation, tests, and documentation based on feedback.
- Milestone 8: Merge changes from Incubator to Core.

7. Class Index

- Zend_File_Convert (factory class)
- Zend_File_Converter (abstract class)
- Zend_File_Convert_Exception

8. Use Cases

UC-01

... (see good use cases book)

9. Class Skeletons

```
class Zend_File_Convert{

   private $_file = array();
   private $_converters = array();

   public __construct($filePath) {
      // Automatic detection of file mime-type ...
   };

   // Can specify directly which file converter to use
   public convertUsing($converter Zend_File_Converter) {
   }

   // Can specify which file converter to use
   public convert($method, $outputFile = NULL) {
       // Loop through converters and pick who can do the conversion
   }

   // Adds a converter
   public add($converter Zend_File_Converter) {
   }
}

abstract class Zend_File_Converter {
  // Supported file conversions (i.e. XHTMLtoPDF)
  abstract public getSupportedMethods(){};
  abstract public isSupportedMethod($file, $method){};
  abstract public convert($file, $outputFile, $method) {};
}

class Zend_File_Convert_Exception extends Zend_Exception {}
```

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