Zend_Filter_Minify_Html - Nick Daugherty

<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_Minify_Html Component Proposal

Proposed Component Name	Zend_Filter_Minify_Html
Developer Notes	http://framework.zend.com/wiki/display/ZFDEV/Zend_Filter_Minify_Html
Proposers	Nick Daugherty
Zend Liaison	TBD
Revision	1.0 - 23 April 2010: Initial Draft. (wiki revision: 15)

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_Minify_Html is a filter to reduce the size of Html strings and files, resulting in faster client downloads

2. References

3. Component Requirements, Constraints, and Acceptance Criteria

- This component **will** minify Html code.
- This component **will** accept both filenames and strings as input to the filter() method.
- This component **will** support writing the minified Html to a file.
- This component **will** use Zend_Filter_Minify_Css and Zend_Filter_Minify_Javascript for <style> and <script> tags
- This component **will** implement Zend_Filter_Interface
- This component **will not** generate invalid Html
- This component **will** provide options for control of minification (such as opting not to remove comments, for example)

4. Dependencies on Other Framework Components

- Zend_Filter
- Zend_Filter_Exception
- Zend_Filter_Minify_Css
- Zend_Filter_Minify_Javascript

5. Theory of Operation

Minifying html leads to faster load times on the client by reducing the overall size of html code. This component aims to perform the work of removing comments, extra whitespace, etc while retaining the familiar Zend_Filter interface.

Zend_Filter_Minify_Html will use the ZF Css and Javascript minifier components to handle any internal <script> and <style> tags encountered when performing minification. This will ensure that all Javascript and Css is kept valid and the resulting html is as minified as possible.

The component can be used by other ZF components, such as Zend_View itself, to (optionally) automatically minify html (and js/css) when rendering output.

The component is used like any other Zend_Filter_* component, passing the string (or filename) to be minified to the filter() method.

6. Milestones / Tasks

- Milestone 1: design notes will be published here
- Milestone 2: Working prototype checked into the incubator supporting minification of Html strings
- Milestone 3: Working prototype checked into the incubator supporting minification of Html files (and optionally writing output to a file)
- Milestone 4: Unit tests exist, work, and are checked into SVN.
- Milestone 5: Initial documentation exists.

7. Class Index

- Zend_Filter_Minify_Exception
- Zend_Filter_Minify_Html

8. Use Cases

UC-01

```
$filter = new Zend_Filter_Minify_Html();

$minified = $filter->filter('<div id="some_html_to_be_minified">     </div>');
```

UC-02

```
$filter = new Zend_Filter_Minify_Html();

$minified = $filter->filter('/path/to/my/file.html');
```

UC-03

This filter could be implemented in a late running plugin or as on option for Zend_View and layouts to automatically minify Html before sending it to the client.

9. Class Skeletons

```


class Zend_Filter_Minify_Exception extends Zend_Filter_Exception {}


/**
 * Zend Framework
 *
 * LICENSE
 *
 * This source file is subject to the new BSD license that is bundled
 * with this package in the file LICENSE.txt.
 * It is also available through the world-wide-web at this URL:
 * http://framework.zend.com/license/new-bsd
 * If you did not receive a copy of the license and are unable to
 * obtain it through the world-wide-web, please send an email
 * to license@zend.com so we can send you a copy immediately.
 *
 * @category   Zend
 * @package    Zend_Filter
 * @copyright  Copyright (c) 2005-2010 Zend Technologies USA Inc. (http://www.zend.com)
 * @license    http://framework.zend.com/license/new-bsd     New BSD License
 * @version    $Id: Int.php 20096 2010-01-06 02:05:09Z nd987 $
 */


/**
 * @see Zend_Filter_Interface
 */
require_once 'Zend/Filter/Interface.php';


/**
 * Based on Minify by Stephen Clay
 *
 * @category   Zend
 * @package    Zend_Filter
 * @copyright  Copyright (c) 2005-2010 Zend Technologies USA Inc. (http://www.zend.com)
 * @license    http://framework.zend.com/license/new-bsd     New BSD License
 */
class Zend_Filter_Minify_Html implements Zend_Filter_Interface
{
    protected $_isXhtml = null;
    protected $_replacementHash = null;
    protected $_placeholders = array();

    /**
     * Defined by Zend_Filter_Interface
     *
     * Returns a string containing the minified Html. Accepts a string
     * or filename of html to be minified
     *
```

```
     * @param  string $value
     * @return string
     */
    public function filter($value)
    {
        //Was the passed $value a file?
        if(is_file($value)){
            $value = file_get_contents($value);
        }

        //Perform minification on $value
        $value =& $this->_minify($value);

        //Return the minified string
        return $value;
    }

    /**
     * Take a string of html and minify it
     *
     * Also takes into account the set options when performing minification
     *
     * If the processor encounters <script> or <style> tags with internal js/css,
     * it will use Zend_Filter_Minify_Css and Zend_Filter_Minify_Javascript to
     * correctly minify css and javascript. Unless option to disable this is set
     *
     * @param  string $html
     * @return string
     */
    protected function _minify($html){
        if ($this->_isXhtml === null) {
            $this->_isXhtml = (false !== strpos($html, '<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML'));
        }

        $this->_replacementHash = 'MINIFYHTML' . md5($_SERVER['REQUEST_TIME']);
        $this->_placeholders = array();

        // replace SCRIPTs (and minify) with placeholders
        $html = preg_replace_callback(
            '/(\\s*)(<script\\b[Zend_Filter_Minify_Html - Nick
Daugherty^>]*?>)([\\s\\S]*?)<\\/script>(\\s*)/i'
            ,array($this, '_removeScriptCB')
            ,$html);

        // replace STYLEs (and minify) with placeholders
        $html = preg_replace_callback(
            '/\\s*(<style\\b[Zend_Filter_Minify_Html - Nick
Daugherty^>]*?>)([\\s\\S]*?)<\\/style>\\s*/i'
            ,array($this, '_removeStyleCB')
            ,$html);

        // remove HTML comments (not containing IE conditional comments).
        $html = preg_replace_callback(
            '/<!--([\\s\\S]*?)-->/'
            ,array($this, '_commentCB')
            ,$html);

        // replace PREs with placeholders
        $html = preg_replace_callback('/\\s*(<pre\\b[Zend_Filter_Minify_Html - Nick
Daugherty^>]*?>[\\s\\S]*?<\\/pre>)\\s*/i'
            ,array($this, '_removePreCB')
            ,$html);

        // replace TEXTAREAs with placeholders
        $html = preg_replace_callback(
            '/\\s*(<textarea\\b[Zend_Filter_Minify_Html - Nick
Daugherty^>]*?>[\\s\\S]*?<\\/textarea>)\\s*/i'
            ,array($this, '_removeTextareaCB')
            ,$html);
```

```
        // trim each line.
        // @todo take into account attribute values that span multiple lines.
        $html = preg_replace('/^\\s+|\\s+$/m', '', $html);

        // remove ws around block/undisplayed elements
        $html = preg_replace('/\\s+(<\\/?(?:area|base(?:font)?|blockquote|body'
            .'|caption|center|cite|col(?:group)?|dd|dir|div|dl|dt|fieldset|form'
            .'|frame(?:set)?|h[1-6]|head|hr|html|legend|li|link|map|menu|meta'
            .'|ol|opt(?:group|ion)|p|param|t(?:able|body|head|d|h||r|foot|itle)'
            .'|ul)\\b[Zend_Filter_Minify_Html - Nick Daugherty^>]*>)/i', '$1', $html);

        // remove ws outside of all elements
        $html = preg_replace_callback(
            '/>([Zend_Filter_Minify_Html - Nick Daugherty^<]+)</'
            ,array($this, '_outsideTagCB')
            ,$html);

        // use newlines before 1st attribute in open tags (to limit line lengths)
        $html = preg_replace('/(<[a-z\\-]+)\\s+([Zend_Filter_Minify_Html - Nick Daugherty^>]+>)/i',
"$1\n$2", $html);

        // fill placeholders
        $html = str_replace(
            array_keys($this->_placeholders)
            ,array_values($this->_placeholders)
            ,$html
        );

        return $html;
    }

    protected function _commentCB($m)
    {
        return (0 === strpos($m[1], '[') || false !== strpos($m[1], '<![']))
            ? $m[0]
            : '';
    }

    protected function _reservePlace($content)
    {
        $placeholder = '%' . $this->_replacementHash . count($this->_placeholders) . '%';
        $this->_placeholders[$placeholder] = $content;
        return $placeholder;
    }

    protected function _outsideTagCB($m)
    {
        return '>' . preg_replace('/^\\s+|\\s+$/', ' ', $m[1]) . '<';
    }

    protected function _removePreCB($m)
    {
        return $this->_reservePlace($m[1]);
    }

    protected function _removeTextareaCB($m)
    {
        return $this->_reservePlace($m[1]);
    }

    protected function _removeStyleCB($m)
    {
        $openStyle = $m[1];
        $css = $m[2];
        // remove HTML comments
        $css = preg_replace('/(?:^\\s*<!--|-->\\s*$)/', '', $css);

        // remove CDATA section markers
        $css = $this->_removeCdata($css);
```

```
        // minify
        if(1==1){
            require_once 'Zend/Filter/Minify/Css.php';

            $filter = new Zend_Filter_Minify_Css();

            $css = $filter->filter($css);
        } else {
            $css = trim($css);
        }

        return $this->_reservePlace($this->_needsCdata($css)
            ? "{$openStyle}/*<![CDATA[*/{$css}/*]]]><![CDATA[>*/</style>"
            : "{$openStyle}{$css}</style>"
        );
    }

    protected function _removeScriptCB($m)
    {
        $openScript = $m[2];
        $js = $m[3];

        // whitespace surrounding? preserve at least one space
        $ws1 = ($m[1] === '') ? '' : ' ';
        $ws2 = ($m[4] === '') ? '' : ' ';

        // remove HTML comments (and ending "//" if present)
        $js = preg_replace('/(?:^\\s*<!--\\s*|\\s*(?:\\/\\/)?\\s*-->\\s*$)/', '', $js);

        // remove CDATA section markers
        $js = $this->_removeCdata($js);

        // minify
        if(1==1){
            require_once 'Zend/Filter/Minify/Javascript.php';

            $filter = new Zend_Filter_Minify_Javascript();

            $js = $filter->filter($js);
        } else {
            $js = trim($js);
        }

        return $this->_reservePlace($this->_needsCdata($js)
            ? "{$ws1}{$openScript}/*<![CDATA[*/{$js}/*]]]><![CDATA[>*/</script>{$ws2}"
            : "{$ws1}{$openScript}{$js}</script>{$ws2}"
        );
    }

    protected function _removeCdata($str)
    {
        return (false !== strpos($str, '<![CDATA['))
            ? str_replace(array('<![CDATA[', ']]]><![CDATA[>'), '', $str)
            : $str;
    }

    protected function _needsCdata($str)
    {
```

```
        return ($this->_isXhtml && preg_match('/(?:[<&]|\\-\\-|\\]\\]>)/', $str));
    }
}
```

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