Zend_Controller_Action_Helper_Autocomplete

<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_Controller_Action_Helper_Autocomplete Component Proposal

Proposed Component Name	Zend_Controller_Action_Helper_Autocomplete
Developer Notes	http://framework.zend.com/wiki/display/ZFDEV/Zend_Controller_Action_Helper_Autocomplete
Proposers	Matthew Weier O'Phinney
Revision	0.8.0 - 08 January 2008: Initial proposal creation 0.9.0 - 09 January 2008: Updated with feedback from Zend team and 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
Zend_Controller_Action_Helper_Autocomplete_Abstract

1. Overview

Zend_Controller_Action_Helper_Autocomplete is intended to automate creation and transmittal of a response for use in client-side autocompletion.

2. References

In no particular order:

- Ajax.Autocompleter in Scriptaculous
- Autocompletion in YUI
- Autocompletion in Dojo
- Autocompletion in JQuery

3. Component Requirements, Constraints, and Acceptance Criteria

- This component **will** provide an abstract class for handling autocomplete resultsets.
- This component **will** provide one or more concrete implementations for returning autocomplete resultsets to specific client-side JS libraries.
- This component **will** set appropriate headers and the response body with the resultset.

4. Dependencies on Other Framework Components

- Zend_Controller_Action_Helper_Abstract
- Zend_Controller_Response_Abstract (for setting response headers and content)
- Zend_Controller_Action_Helper_Json (for serializing JSON resultsets and setting response headers)

5. Theory of Operation

Autocompletion allows type-ahead type functionality for HTML text input, providing users with options that may match the input they've typed. There are many ways to provide such functionality on the server-side, including:

- Returning HTML unordered lists (Prototype/Scriptaculous)
- Returning JSON arrays-of-arrays (YUI)
- Returning JSON objects (basically arrays-of-arrays) (Dojo)
- Returning JSON array-of-strings (JQuery)

Due to the variety of return types, no one component may serve all scenarios. However, the base functionality is relatively the same: take an array or iterable resultset of data, and transform it to the appropriate return value.

The Autocomplete helper will provide an abstract class for the base operations, including serializing the data to JSON return values. Concrete classes will thenformat the return value for the appropriate library.

6. Milestones / Tasks

- Milestone 1: [DONE] Complete proposal
- Milestone 2: Write tests and code and publish to incubator
- Milestone 3: Write documentation and use cases
- Milestone 4: Publish to core

7. Class Index

- Zend_Controller_Action_Helper_AutoComplete_Abstract
- Zend_Controller_Action_Helper_AutoCompleteDojo
- Zend_Controller_Action_Helper_AutoCompleteScriptaculous
- Zend_Controller_Action_Helper_AutoCompleteJquery

8. Use Cases

UC-01

```
class FooController extends Zend_Controller_Action
{
    public function barAction()
    {
        // ...
        if ($this->_helper->ajaxContext->getCurrentContext()) {
            // Return autocompletion results for Dojo:
            return $this->_helper->autoCompleteDojo($data);
        }
        // ...
    }

    public function bazAction()
    {
        // ...
        if ($this->_helper->ajaxContext->getCurrentContext()) {
            // Return autocompletion results for Scriptaculous:
            return $this->_helper->autoCompleteScriptaculous($data);
        }
        // ...
    }
}
```

9. Class Skeletons

Zend_Controller_Action_Helper_Autocomplete_Abstract

```
class Zend_Controller_Action_Helper_AutoComplete_Abstract extends
Zend_Controller_Action_Helper_Abstract
{
    /**
     * Encode data as JSON and set headers
     */
    public function encodeJson($data);

    /**
     * Strategy pattern: proxy to appropriate encoding
     */
    abstract public function direct($data);
}
```

```
class Zend_Controller_Action_Helper_AutoCompleteDojo extends
Zend_Controller_Action_Helper_AutoComplete_Abstract
{
    public function direct($data); // proxies to encodeJson()
}
```

```
class Zend_Controller_Action_Helper_AutoCompleteScriptaculous extends
Zend_Controller_Action_Helper_AutoComplete_Abstract
{
    public function direct($data); // generates response
}
```

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