# Zend_Controller_Response output buffering

<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_Request Output Buffering Component Proposal

| | |
|---|---|
| **Proposed Component Name** | Zend_Controller_Request Output Buffering |
| **Developer Notes** | http://framework.zend.com/wiki/display/ZFDEV/Zend_Controller_Request Output Buffering |
| **Proposers** | Matthew Weier O'Phinney |
| **Revision** | 1.0 - 3 March 2007: Initial proposal (wiki revision: 4) |

## 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_Controller_Response should be aggregate all responses emitted by the controller chain. This has the benefit of ensuring that headers are always emitted prior to content, as well as allowing the developer to write unit tests against their front controller. A simple way to do so would be to use output buffering.

## 2. References

- PHP Manual: output buffering
- Zend_Controller_Response documentation

## 3. Component Requirements, Constraints, and Acceptance Criteria

- Output buffering **will** happen at the dispatcher level, surrounding the call to an action method
- Content captured by output buffering **will** be **appended** to the body content of the response object
- Developers **will** be able to turn off the output buffering feature via a front controller parameter

## 4. Dependencies on Other Framework Components

- Zend_Controller_Response_Abstract
- Zend_Controller_Dispatcher_Standard
- Zend_Controller_Front

## 5. Theory of Operation

This feature will be on by default and transparent to the end-user.

## 6. Milestones / Tasks

- Milestone 1: proposal
- Milestone 2: working code committed to the repo and tested
- Milestone 3: documentation of the feature

## 7. Class Index

- Zend_Controller_Dispatcher_Standard

## 8. Use Cases

**UC-01**

```
// sample controller that echoes data
class IndexController extends Zend_Controller_Action
{
    public function indexAction()
    {
        echo "In the index action of the index controller";
    }
}

// in the bootstrap
$front->returnResponse(true);
$response = $front->dispatch();

// echoes "In the index action of the index controller"
echo $response->getBody();
```

**UC-02**

```
// same controller/action as above
$front->setParam('disableOutputBuffering', true);
$front->returnResponse(true);
$response = $front->dispatch();

// empty string
echo $response->getBody();
```

## 9. Class Skeletons

```
class Zend_Controller_Dispatcher_Standard extends Zend_Controller_Dispatcher_Abstract
{
    public function dispatch(Zend_Controller_Request_Abstract $request,
Zend_Controller_Response_Abstract $response)
    {
        // ... determine controller and action, etc

        // dispatching the method call:
        $disableOb = $this->getParam('disableOutputBuffering');
        $request->setDispatched(true);
        if ($disableOb) {
            $controller->dispatch($action);
        } else {
            ob_start();
            $controller->dispatch($action);
            $content = ob_get_flush();
            if (!empty($content)) {
                $response->appendBody($content);
            }
        }

        // continue as normal...
    }
}
```

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