**Controller Overview**

**Zend\_Controller\_Front:**

* Zend\_Controller\_Front manage the entire workflow of the Zend\_Controller system.
* Zend\_Controller\_Front processes all requests received by the server and is ultimately responsible for represent requests to ActionControllers (Zend\_Controller\_Action).

**Zend\_Controller\_Request\_Abstract:**

* Zend\_Controller\_Request\_Abstract represents the request environment and provides methods for setting and retrieving the controller and action names and any request parameters.

**Zend\_Controller\_Router\_Interface:**

* Routing is the process of examining the request environment to determine which controller, and action of that controller, should receive the request.
* Zend\_Controller\_Router\_Interface is used to define routers.
* Routing occurs only once, when the request is initially received and before the first controller is dispatched.
* The default router, Zend\_Controller\_Router\_Rewrite, takes a URI endpoint as specified in Zend\_Controller\_Request\_Http and decomposes it into a controller, action, and parameters based on the path information in the URL.

**Zend\_Controller\_Dispatcher\_Interface:**

* Zend\_Controller\_Dispatcher\_Interface is used to define dispatchers.
* Dispatching is the process of taking the request object, Zend\_Controller\_Request\_Abstract, extracting the module name, controller name, action name, and optional parameters contained in it, and then instantiating a controller and calling an action of that controller.
* If any of the module, controller, or action are not found, it will use default values for them.
* Zend\_Controller\_Dispatcher\_Standard specifies index for each of the controller and action defaults, but allows the developer to change the default values for each using the setDefaultController(), setDefaultAction(), and setDefaultModule() methods, respectively.

**Zend\_Controller\_Action:**

* Zend\_Controller\_Action is the base action controller component.
* Each controller is a single class that extends the Zend\_Controller\_Action class and should contain one or more action methods.

**Zend\_Controller\_Response\_Abstract:**

* Zend\_Controller\_Response\_Abstract is used to return response to the view and it collect both body and header contents.
* The default response class is Zend\_Controller\_Response\_Http, which is suitable for use in an HTTP environment.

**The Front Controller**

Zend\_Controller\_Front purpose is to initialize the request environment, route the incoming request, and then dispatch any discovered actions.

In Zend\_Controller\_Front only a single instance of it may be available at any given time.

Zend\_Controller\_Front registers a plugin broker with itself, allowing various events it triggers to be observed by plugins. By default, the front controller loads the ErrorHandler plugin, as well as the ViewRenderer action helper plugin. To disable the ErrorHandler, perform the following at any point prior to calling dispatch().

$front->setParam('noErrorHandler', **true**);

$front->setParam('noViewRenderer', **true**);

## **Primary Methods**

### **getInstance() :**

getInstance() is used to retrieve a front controller instance.

$front = Zend\_Controller\_Front::getInstance();

**setControllerDirectory() :**

setControllerDirectory() is used to tell the dispatcher where to look for action controller class files.

$front->setControllerDirectory('../application/controllers');

$front->setControllerDirectory([array](http://www.php.net/array)(

'default' => '../application/controllers'

'blog'    => '../modules/blog/controllers',

'news'    => '../modules/news/controllers',

));

### **addModuleDirectory()**

### **getModuleDirectory()**

### **dispatch() :**

dispatch(

Zend\_Controller\_Request\_Abstract $request,

Zend\_Controller\_Response\_Abstract $response

);

Both request and response objects are optional.

The dispatch process has three distinct events:

* Routing
* Dispatching
* Response

### **run() :**

Simply it fetches a front controller instance, registers the path provided via setControllerDirectory(), and finally dispatches.

Zend\_Controller\_Front::run('../application/controllers');

## **Environmental Accessor Methods**

* resetInstance()
* setDefaultControllerName()
* getDefaultControllerName()
* setDefaultAction()
* getDefaultAction()
* setRequest()
* setResponse()
* setDispatcher()
* setRouter()

## **Front Controller Parameters**

* setParams(array $params)
* setParam($name, $value)
* getParam($name)
* getParams()
* clearParams()

## **Extending the Front Controller**

class My\_Controller\_Front extends Zend\_Controller\_Front

{

    public [static](http://www.php.net/static) function getInstance()

    {

        if (null === self::$\_instance) {

            self::$\_instance = new self();

        }

        return self::$\_instance;

    }

}

Overriding the getInstance() method ensures that subsequent calls to Zend\_Controller\_Front::getInstance() will return an instance of your new subclass instead of a Zend\_Controller\_Front instance

# The Request Object

# The request object is passed between Zend\_Controller\_Front and the router, dispatcher, and controller classes.

# The module name is accessed by getModuleName() and setModuleName().

# The controller name is accessed by getControllerName() and setControllerName().

# The name of the action to call within that controller is accessed by getActionName() and setActionName().

### **HTTP Requests - Accessing Request Data**

Zend\_Controller\_Request\_Http used to access to all values such as the key name and value for the controller and action router variables, and all additional parameters parsed from the URI.

### **Determining the Request Method**

getMethod() allows you to determine the HTTP request method used to request the current resource.

There is many other methods to get values from the request,

 isGet()

 isPost()

 isPut()

# The Standard Router

Zend\_Controller\_Router\_Rewrite is the standard framework router and Routing occurs only once before dispatch.

Routing is the process of taking a URI endpointand decomposing it into parameters to determine which module, controller, and action of that controller should receive the request.

This values of the module, controller, action and other parameters are packaged into a Zend\_Controller\_Request\_Http object.

Module + controller + action + params:

http:*//example/blog/archive/list/sort/alpha/date/desc*

## **Using a Router ( Create ):**

$router = $ctrl->getRouter(); *// returns a rewrite router by default*

$router->addRoute(

'user',

**new** Zend\_Controller\_Router\_Route('user/:username',

[array](http://www.php.net/array)('controller' => 'user',

'action' => 'info'))

);

**Types of routes:**

 Zend\_Controller\_Router\_Route

 Zend\_Controller\_Router\_Route\_Static

 Zend\_Controller\_Router\_Route\_Regex

 Zend\_Controller\_Router\_Route\_Hostname

 Zend\_Controller\_Router\_Route\_Chain

 Zend\_Controller\_Router\_Rewrite

# The Dispatcher

# Dispatching is the process of taking the request object, Zend\_Controller\_Request\_Abstract, extracting the module name, controller name, action name, and optional parameters contained in it, and then instantiating a controller and calling an action of that controller.

If any of the module, controller, or action are not found, it will use default values for them. Zend\_Controller\_Dispatcher\_Standard specifies index for each of the controller and action defaults and default for the module default value, but allows the developer to change the default values for each using the setDefaultController(), setDefaultAction(), and setDefaultModule() methods, respectively.

1. **public** **function** oneAction()
2. {
3. $this->\_forward('main', **null**, **null**, [array](http://www.php.net/array)('baz' => 'bogus'));
4. }
5. **public** **function** twoAction()
6. {
7. $this->\_forward('main', 'sub', **null**, [array](http://www.php.net/array)('baz' => 'bogus'));
8. }
9. **public** **function** threeAction()
10. {
11. $this->\_forward('main', 'sub', 'tag', [array](http://www.php.net/array)('baz' => 'bogus'));
12. }

**Zend\_Controller\_Dispatcher\_Interface:**

Zend\_Controller\_Dispatcher\_Interface defines the following methods as required for any dispatcher implementation:

* isDispatchable()
* setParam()
* getParam()
* clearParams()
* setResponse()

# Action Controllers