Basic Application using Zend Framework

@author: Md. Mahmud Ahsan
url: http://mahmudahsan.wordpress.com

Introduction:

Once upon a time, we develop web application using raw level php code. But when the application expand it becomes unmanageable. Now it is a trend to develop web application using a framework. And I found Zend Framework is one of the best framework among all. Here I given a short tutorial to develop web application using Zend framework. I assume that, the reader knows what is MVC, how to setup apache server, mysql, php, how to enable mod_rewrite etc. This is a basic tutorial and for complete reference download manual from http://framework.zend.com/

Directory Structure:

First download the framework from http://framework.zend.com/ . Current version is 1.5.2. After download, unzip the file.

Suppose your application name: mysite . Then setup the directory structure like this

mysite/
- /application
- /library
- /public

Here library is the folder copied from unzipped folder of Zend framework. In library folder you'll see a folder named Zend and it contains all the essentials of this framework.

Now structure your application folder

application/
- /config
- /controllers
- /models
- /views/scripts
- /layouts

public/
- /css
- /js
- /images

For security reason, you should place the resource files like images, css, js in separate.
Now create a file config.ini and place it in your mysite/application/config folder.

Filename: config.ini

```
[general]
host                =   http://localhost/mysite
db.adapter          =   PDO_MYSQL
db.params.host      =   localhost
db.params.username  =   root
db.params.password  =   xxxxxx
db.params.dbname    =   mysite_db
```

Here we provide the hostname of our site and database configuration. If you don't need database, just remove from db.adapter to db.params.dbname from config.ini

Now create a .htaccess file and place it in your mysite folder.

Filename: .htaccess

```
# Rewrite rules for Zend Framework
RewriteEngine on
RewriteCond %{REQUEST_FILENAME} !-f
RewriteRule .* index.php

# Security: Don't allow browsing of directories
Options -Indexes
```

Now create an index.php file and place it to your mysite folder.

Filename: index.php

```
<?php
    error_reporting(E_ALL);
    ini_set('display_errors', 1);

    // directory setup and class loading
    set_include_path('.' . PATH_SEPARATOR . 'library/'
         . PATH_SEPARATOR . 'application/models'
         . PATH_SEPARATOR . 'application/config'
         . PATH_SEPARATOR . get_include_path());

    include "Zend/Loader.php";

    Zend_Loader::registerAutoload();

    //load configuration
    $config         =   new Zend_Config_Ini('application/config/config.ini', 'general');
    $registry       =   Zend_Registry::getInstance();
    $registry->set('config', $config);

    //setup database
    $db             =  Zend_Db::factory($config->db);
    Zend_Db_Table::setDefaultAdapter($db);
    $registry->set('db', $db);

    // setup controller
    $frontController = Zend_Controller_Front::getInstance();
    $frontController->throwExceptions(true);
    $frontController->setControllerDirectory('application/controllers');
    Zend_Layout::startMvc(array('layoutPath'=>'application/layouts'));

    // run!
    $frontController->dispatch();

?>
```

Zend Framework is designed such that its files must be on the include path. By set_include_path(.....) we include some common paths. And for this reason, include “Zend/Loader.php” automatically include Loader.php from /library folder.

`Zend_Loader::registerAutoload()` automatically load all Zend Framework files as we instantiate them.

Zend_Registry::getInstance() returns a registry object. We can store any value, object in it. It is like a global scope. And in anywhere in our model or controller we can simply retrive the value by Zend_Registry::get('variable_name')

Now its the time to create a controller file. And put it to the controllers folder

```
filename: IndexController.php

class IndexController extends Zend_Controller_Action{

        function init(){
             //setup the common tasks
        }
        function indexAction(){
            $this->view->title              =   "My Site";
        }

        function showAction(){
             $this->view->title              =   "My Site data";
            $this->view->data               =   array('name'=> Mahmud, 'work'=> “BD”);
        }
}

To create an Action Controller remember:
• Extend Zend_Controller_Action
• Class name ends in 'Controller'
     IndexController
     BlogController

To create a Controller Action remember:
• Public methods ending in 'Action'
    barAction()
    indexAction()
```

Now the time to create view files.
Create a folder named index and put it to /views/scripts/ . So the path is: /views/script/index/

If we create a controller named BlogController , we have to create view folder like this way and the folder name will be blog and have to put in /views/scripts/blog/

In /views/script/index/ , create files named index.phtml and show.phtml

filename: index.phtml

```
<h1><?=$this->title?></h1>
```

filename: show.phtml

```
<span>
    <table>
    <?php
        foreach($this->data as $key=>$value){
            echo "<tr><td>$key: </td> <td>$value</td></tr>";
        }
    ?>
    </table>
</span>
```

For common html code, like common header, footer or navigation bar, we have to create a template in

/application/layouts/
Now create a file layout.phtml and put it in /application/layouts/layout.phtml

Filename: layout.phtml

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en" lang="en">
<head>
    <meta http-equiv="Content-Type" content="text/html;charset=utf-8" />
    <title><?php echo $this->title; ?></title>
</head>
<body>
<div id="content">
    <?php echo $this->layout()->content; ?>
</div>
</body>
</html>
```

Now in your browser, run this url. http://localhost/mysite/index
You'll see only title of the application is shown.

Now run this url: http://localhost/mysite/index/show
You'll see the data and title.

So what happened when you run localhost/mysite/index
It points to IndexController and index action (default action)

In the second case, in points IndexController and show action.

After completing the action method of controller it automatically render the corresponding view file.
For showAction() the corresponding view file is /views/scripts/index/show.phtml
And the output is inserted in the /application/layouts/layout.phtml in `<?php echo $this->layout()->content; ?>`

If you want to create a model class that handles all the database operation. Here is an example. Create a file Test.php and put it in application/models/Test.php

Filename: Test.php

```
<?php
    class Test{
        private $db;

        function __construct(){
            $this->db           =   Zend_Registry::get('db');
        }
        function show(){
            $sql    =   "SELECT * from temp";
            $result =   $this->db->fetchAll($sql);
            return $result;
        }
    }
?>
```

At first create a database named: test. Now create a table named temp. temp table contains only id,

name

create table temp{
	id int not null auto_increment,
	name varchar(40),
	primary key (id)
}

Insert some data.

Now in your IndexController, showAction() method write this

```
$test   =   new Test();
$result =   $test->show();

echo “<pre>”;
print_r($result);
```

There are many ways to handling database. I've shown one of these. Hope this will be helpful.

Conclusion: I compose this tutorial within very short time. I didn't explain everything in details for my busy work. For complete reference download the documentation from http://framework.zend.com/