Zend Framework

Subjects: Computer Science, Software Engineering
Contributor: HandWiki Zheng

Zend Framework (ZF) is an open source, object-oriented web application framework implemented in PHP 7 and licensed under the New BSD License. The framework is basically a collection of professional PHP-based packages. The framework uses various packages by the use of Composer as part of its package dependency managers; some of them are PHPUnit for testing all packages, Travis CI for continuous Integration Services. Zend Framework provides to users a support of the Model View Controller (MVC) in combination with Front Controller solution. MVC implementation in Zend Framework has five main areas. The router and dispatcher functions to decide which controller to run based on data from URL, and controller functions in combination with the model and view to develop and create the final web page. On 17 April 2019 it was announced that the framework is transitioning into an open source project hosted by the Linux Foundation to be known as Laminas.

Keywords: open source project ; web application ; model

1. License

Zend Framework is licensed under the Open Source Initiative (OSI)-approved New BSD License. For ZFv1 all code contributors must sign a Contributor License Agreement (CLA) based on the Apache Software Foundation's CLA. The licensing and contribution policies were established to prevent intellectual property issues for commercial ZF users, according to Zend's Andi Gutmans.[1] ZF2 and later is CLA free.[2] There is also a longterm support available for the framework (long term support or LTS) for a total duration of 3 years.

2. Zend Framework Components and Versioning

Starting with Zend Framework version 2.5, components are split into independently versioned packages and zendframework/zendframework is converted into a Composer meta-package. Framework components introduced after the split are not added to the meta-package.

While zendframework/zendframework meta-package release version remains at 3.0.0, it will instruct Composer to install latest compatible versions of the framework components, as per the semantic versioning. Such that zend-mvc component will be installed at its current version 3.1.1, zend-servicemanager at version 3.3.0 and zend-form at version 2.10.2.

Zend Framework includes following components:[3]

Authentication	**Authenticate users via a variety of adapters, and provide the authenticated identity to your application.**
Barcode	**Programmatically create and render barcodes as images or in PDFs.**
Cache	**Caching implementation with a variety of storage options, as well as codified caching strategies for callbacks, classes, and output.**
Captcha	**Generate and validate CAPTCHAs using Figlets, images, ReCaptcha, and more.**
Code	**Extensions to the PHP Reflection API, static code scanning, and code generation.**
Component Installer	**Composer plugin for injecting modules and configuration providers into application configuration.**
Config	**Read and write configuration files.**
Config Aggregator	**Aggregate and merge configuration from a variety of sources.**
Console	**Build console applications using getopt syntax or routing, complete with prompts**
Crypt	**Strong cryptography tools and password hashing.**

DB	**Database abstraction layer, SQL abstraction, result set abstraction, and RowDataGateway and TableDataGateway implementations.**
Debug	**Safely dump debug information to HTML.**
DI	**Automated dependency injection and instance manager.**
Diactoros	**PSR-7 HTTP message implementations.**
DOM	**Query HTML and XML documents using XPath or CSS selectors.**
Escaper	**Securely and safely escape HTML, HTML attributes, JavaScript, CSS, and URLs.**
EventManager	**Implement events, signal slots, aspects, and observers!**
Expressive	**PSR-7 middleware in minutes.**
Feed	**Consume and generate Atom and RSS feeds, and interact with Pubsubhubbub.**
File	**Locate PHP classfiles.**
Filter	**Programmatically filter and normalize data and files.**
Form	**Validate and display simple and complex forms, casting forms to business objects and vice versa.**
HAL for PSR-7	**Hypertext Application Language (HAL) for PSR-7.**
HTTP	**HTTP message and header abstractions, and HTTP client implementation. (Not a PSR-7 implementation.)**
Hydrator	**Serialize objects to arrays, and vice versa.**
InputFilter	**Normalize and validate input sets from the web, APIs, the CLI, and more, including files.**
Internationalization	**Provide translations for your application, and filter and validate internationalized values.**
JSON	**De/Serialize JSON in PHP, including JavaScript expressions.**
JSON-RPC Server	**JSON-RPC implementation for PHP.**
LDAP	**Perform LDAP operations, including binding, searching and modifying entries in an LDAP directory.**
Loader	**Autoloading and plugin loading strategies.**
Log	**Robust, composite logger with filtering, formatting, and PSR-3 support.**
Mail	**Parse, create, store, and send email messages, using a variety of storage and transport protocols.**
Math	**Create cryptographically secure pseudo-random numbers, and manage big integers.**
Memory	**Manage data in an environment with limited memory.**
MIME	**Create and parse MIME messages and parts.**
Module Manager	**Modular application system for zend-mvc applications.**
MVC	**Zend Framework's event-driven MVC layer, including MVC Applications, Controllers, and Plugins.**
MVC-Console integration	**Integration between zend-mvc and zend-console.**
MVC-i18n integration	**Integration between zend-mvc and zend-i18n.**
fileprg() plugin	**Post/Redirect/Get plugin with file upload handling for zend-mvc controllers.**
flashmessenger() plugin	**Plugin for creating and exposing flash messages via zend-mvc controllers.**
identity() plugin	**Plugin for retrieving the current authenticated identity within zend-mvc controllers.**
prg() plugin	**Post/Redirect/Get plugin for zend-mvc controllers.**
Navigation	**Manage trees of pointers to web pages in order to build navigation systems.**
Paginator	**Paginate collections of data from arbitrary sources.**
ACL	**Create, manage, and query access control lists.**

RBAC	**Provide and query Role-Based Access Controls for your application.**
Problem Details	**PSR-7 Problem Details for HTTP API responses and middleware.**
ProgressBar	**Create and update progress bars in different environments.**
PSR-7 Bridge	**PSR-7 <-> zend-http message conversions.**
Router	**Flexible routing system for HTTP and console applications.**
Serializer	**Serialize and deserialize PHP structures to a variety of representations.**
Server	**Create Reflection-based RPC servers.**
ServiceManager	**Factory-Driven Dependency Injection Container**
ServiceManager-Di integration	**zend-di integration for zend-servicemanager**
Session	**Object-oriented interface to PHP sessions and storage.**
SOAP	**Create, serve, and access SOAP applications, and parse and generate WSDL.**
Stdlib	**SPL extensions, array utilities, error handlers, and more.**
Stratigility	**PSR-7 middleware foundation for building and dispatching middleware pipelines.**
Tag	**Manipulate and weight taggable items, and create tag clouds.**
Test	**Tools to facilitate unit testing of zend-mvc applications.**
Text	**Create FIGlets and text-based tables.**
URI	**Object oriented interface to URIs, with facilities for validation.**
Validator	**Validation classes for a wide range of domains, and the ability to chain validators to create complex validation criteria.**
View	**Flexible view layer supporting and providing multiple view layers, helpers, and more.**
XML-RPC	**Fully featured XML-RPC server and client implementations.**
XML2JSON	**Convert XML documents to JSON.**

3. Installation

Officially supported install method is via Composer package manager.

Zend Framework provides meta-package that includes 61 component but recommended way is to install required framework components individually. Composer will resolve and install all additional dependencies.

For instance, if you need MVC package, you can install with the following command:

```
$ composer require zendframework/zend-mvc
```

Full list of components is available in Zend Framework documentation.[3]

4. Anatomy of Zend Framework

Zend Framework follows configuration-over-convention approach and does not impose any particular application structure. Skeleton applications for zend-mvc and zend-expressive are available and provide everything necessary to run applications and to serve as a good starting point.

4.1. Recommended MVC Application Directory Structure

ZendSkeletonApplication, skeleton application using Zend Framework MVC layer and module systems, can be installed with:

```
$ composer create-project zendframework/skeleton-application <project-path>
```

It will create file structure similar to this:

```
<project name>/
```

├── config/ │ ├── autoload/ │ │ ├── global.php │ │ └── local.php.dist │ ├── application.config.php │ └── modules.config.php ├── data/ │ └── cache/ ├── module/ ├── public/ │ └── index.php ├── vendor/ ├── composer.json ├── composer.lock

```
└── phpunit.xml.dist
```

The config/ directory has application wide configurations. module/ directory contains local modules that are committed along with application. vendor/ contains vendor code and other modules managed independently from the application, content of the folder is normally managed by Composer.

Zend Framework module have only one requirement: Module class exists in a module namespace and is autoloadable. Module class provides configuration and initialization logic to application. Recommended module structure is as follows:

```
<modulename> ├── config/ │ └── module.config.php ├── src/ │ └── Module.php ├── test/ ├── view/ ├── composer.json └── phpunit.xml.dist
```

The config/ directory holds module configs, src/ directory contains module source code, as defined in PSR-4 autoloading standard, test/ directory contains unit tests for the module and view/ directory holds view scripts.

5. Creating Project Structure

Zend framework supports command line input to create structure of directories. We will use command line interface to start creating the directory structure for our project. This will give you complete structural understanding of directories. The interface supports and provides Zend_Tool interface giving a whole host of command functionalities.

1. Open the command line interface, and change the hellozend directory.
2. Windows users type: bin\zf\bat create project
3. Linux/Mac users type: bin\zf.sh create project

This procedure will create Zend Framework project in a your own specified location. After running Zend_Toll it will create the basic application skeleton.[4] This will not only create directory structure but also all the basic elements of the MVC framework.[4] In order to get Apache functionalities the virtual host settings will be as:[4]

```
Listen 8080 <VirtualHost *: 8080> DocumentRoot /User/keithpope/Sites/hellozend/public
</VirtualHost>
```

The basic directory structure created will be somewhat as mentioned in the aforementioned directory structure of Zend Framework with similar explanation. There is another aspect of Zend-Tool which is automatically initialized during installation is bootstrapping. Here the basic purpose is to initialize the request of page by developer. The main entry here created by Zend Framework is the Index file. Index file provides function to handle user request. This is the main entry point for all requests. Following shows the functionalities.[4]

1. Application-path: defines the path to application directory
2. Application_Env: changes the application behavior depending on various factors such as how the application is used.
3. getenv(): checks system environment.
4. Initialize Zend-Application application: includes Zend-Application and create an instance of it.
5. Call bootstrap() method coupled with run() method starting MVC.

In general Zend-Tool creates many important directory structures. This system is built upon Rapid Application Development technology. As a general rule of support the framework focuses on coding and project structures instead of focusing on smaller parts.[5]

- Project directory structure
- Controllers
- Actions
- Views
- Bootstrap file

5.1. Controllers

Controller is the main entry to Zend Framework application.[6] The front controller handler is main hub for accepting requests and running the accurate actions as requested by the commands. The whole process of requesting and reacting is routing and dispatching (which basically means calling correct methods in a class) which determines the functionality of the code.[6] This is implemented by Zend_Controller_Router_- Interface.[6] The router functionality is to find which actions need to be run and on contrary dispatcher runs those requested actions.[6] The controller in Zend Framework is connected in a diverse array of structural directories, which provides a support to efficient routing.[6] The main entry point and the command controller is the Zend_Controller_Front, this works as a foundation which delegates the work received and sent. The request is shaped and encapsulated with an instance of Zend Controller Request HTTP, as a provider of access to HTTP requests.[6] The HTTP hold all the superglobals of the framework ($_GET, $_POST, $_COOKIE, $_SERVER, and $_ENV) with their relevant paths. Moreover, the controller also provides getParam() functions which enables collection of requested variables.

5.2. Actions

Actions are important functionalities. Controllers do not function without Actions. For this purpose we create another method which has action appended in its name and automatically the front controller will recognize it as an action.[4] The Action has init() method which shows its private nature and not accessible by anyone.[4] Following commands are run so that Zend_Tool can create action for us.[4] Through the use of standard dispatcher all functions are named after the action's name and followed by word "Action" appended.[6] This leads to controller action class containing methods like indexAction(), viewAction(), editAction(), and deleteAction().

Windows users:

```
bin\zf.bat create actions about index
```

Linux and Mac users:

```
bin/zf.sh create action about index
```

An example of forms and actions:[7]

```
namespace Album\Form; use Zend\Form\Form; class AlbumForm extends Form { public function __construct($name = null) { // we want to ignore the name passed parent::__construct('album'); $this->add(array( 'name' => 'id', 'type' => 'Hidden', )); $this->add(array( 'name' => 'title', 'type' => 'Text', 'options' => array( 'label' => 'Title', ), )); $this->add(array( 'name' => 'artist', 'type' => 'Text', 'options' => array( 'label' => 'Artist', ), )); $this->add(array( 'name' => 'submit', 'type' => 'Submit', 'attributes' => array( 'value' => 'Go', 'id' => 'submitbutton', ), )); } // source: Zend Framework Guide }
```

5.3. Standard Router

Standard router is an important Front Controller tool. Here the main decisions are made in order what module, controller and action are being requested.[4] These are all processed here. The following are defaults structure.

1. Module
2. Controller
3. Actions

The request follows a pattern first information is taken from URL endpoint of HTTP. URI is the end point of the request. URL structure follows as:[4] **http://domain.com/moduleName/controllerName/actionName**

The default router code example:[8]

```
// Assuming the following: $ctrl->setControllerDirectory( array( 'default' => '/path/to/default/controllers', 'news' => '/path/to/news/controllers', 'blog' => '/path/to/blog/controllers' ) );
```

Module only:
http://example/news
module == news

Invalid module maps to controller name:
http://example/foo
controller == foo

Module + controller:
http://example/blog/archive
module == blog
controller == archive

Module + controller + action:
http://example/blog/archive/list
module == blog
controller == archive
action == list

Module + controller + action + params:
http://example/blog/archive/list/sort/alpha/date/desc
module == blog
controller == archive
action == list
sort == alpha
date == desc

5.4. Utility Methods

The Zend Framework also provides some utility methods. Following are some utility methods provided in the framework.[4]

_forward()it is used to call action

```
_forward{$action, $controller = null, $module = null, array $params = null}
```

$actionsstring, action required
$controlleroptional string parameter and is place where controller is in.
$modulestring, has module in which we have the controller.
$paramsarray, user parameter

Another method is the redirect utility method. This is the opposite of aforementioned _forward() method.[4] _redirect() performs HTTP in redirection in creation of a new request.[4] _redirect() methods accepts two arguments namely $url, and $options.

Furthermore, Action Helpers are also a way to provide extra functionalities within the framework. Action helpers are useful when there is a need to provide functionality between controllers.[4]

```
//application/controllers/IndexController.php public function init() { $this->_helper->viewRenderer->setNoRender(); }
```

During initialization phase of IndexController and ContactController, viewReader is called and noRender flag is called on the view object.[4] The lack of this process creates an error in our application.

5.5. View Directories

Zend Framework provides the view framework to our project and controller and actions are automatically provided to our application. Inside the Zend Framework in view folder we observe the following folders.[4]

1. View
2. Helpers
3. Scripts

4. Contacts
5. errors
6. index

In order to create a view we follow:[4]

```
<!-- application/views/scripts/index/index.phtml --> <html> <head> <title><Hello Zend</title> </head> <body> <hi>Hello Zend</hi> <p>Hello from Zend Framework</p> </body> </html>
```

View Sample:[9]

```
// https://framework.zend.com/manual/2.4/en/modules/zend.view.quick-start.html
```

namespace Foo\Controller;

use Zend\Mvc\Controller\AbstractActionController; use Zend\View\Model\ViewModel;

class BazBatController extends AbstractActionController {

```
 public function doSomethingCrazyAction() { $view = new ViewModel(array( 'message' => 'Hello world', )); $view->setTemplate('foo/baz-bat/do-something-crazy'); return $view; }
```

```
}
```

6. Sponsor and Partners

Zend Technologies, co-founded by PHP core contributors Andi Gutmans and Zeev Suraski, is the corporate sponsor of Zend Framework.[10] Technology partners include IBM,[11] Google,[12] Microsoft,[13] Adobe Systems,[14] and StrikeIron.[15]

7. Features

Zend Framework features include:[16]

- All components are fully object-oriented PHP 5 and are E_STRICT compliant, which helps in the development of building tests and writing codes in a bug-free and crash-proof application manner.[17]
- Use-at-will architecture with loosely coupled components and minimal interdependencies
- Extensible MVC implementation supporting layouts and PHP-based templates by default
- Support for multiple database systems and vendors, including MariaDB, MySQL, Oracle, IBM DB2, Microsoft SQL Server, PostgreSQL, SQLite, and Informix Dynamic Server
- Email composition and delivery, retrieval via mbox, Maildir, POP3 and IMAP4
- Flexible caching sub-system with support for many types of backends, such as memory or a file system.
- With the help of remote procedure call (RPC) and REST(Representational State Transfer) services, Zend Apigility helps developers to create APIs, authentication of APIs, documentation of APIs, Easy Modification[18]

8. Development of Applications

Zend Framework applications can run on any PHP stack that fulfills the technical requirements. Zend Technologies provides a PHP stack, Zend Server (or Zend Server Community Edition), which is advertised to be optimized for running Zend Framework applications. Zend Server includes Zend Framework in its installers, along with PHP and all required extensions. According to Zend Technologies, Zend Server provides improved performance for PHP and especially Zend Framework applications through opcode acceleration and several caching capabilities, and includes application monitoring and diagnostics facilities.[19] Zend Studio is an IDE that includes features specifically to integrate with Zend Framework. It provides an MVC view, MVC code generation based on Zend_Tool (a component of the Zend Framework), a code formatter, code completion, parameter assist, and more.[20] Zend Studio is not free software, whereas the Zend Framework and Zend Server Community Edition are free. Zend Server is compatible with common debugging tools such

as Xdebug. Other developers may want to use a different PHP stack and another IDE such as Eclipse PDT which works well together with Zend Server. A pre configured, free version of Eclipse PDT with Zend Debug is available on the Zend web site.

9. Code, Documentation, and Test Standards

Code contributions to Zend Framework are subject to rigorous code, documentation, and test standards. All code must meet ZF's coding standards and unit tests must reach 80% code coverage before the corresponding code may be moved to the release branch.[21]

10. Simple Cloud API

On September 22, 2009, Zend Technologies announced[22] that it would be working with technology partners including Microsoft, IBM, Rackspace, Nirvanix, and GoGrid along with the Zend Framework community to develop a common API to cloud application services called the Simple Cloud API. This project is part of Zend Framework and will be hosted on the Zend Framework website,[23] but a separate site called simplecloud.org[24] has been launched to discuss and download the most current versions of the API.The Simple Cloud API and several Cloud Services are included in Zend Framework. The adapters to popular cloud services have reached production quality.

11. Hello World: File by File

In order to create Hello World program, there are multiple steps including:

- First create four files within the directory structure. These files are bootstrap file, an Apache Control file (.htaccess), a controller file and a view controller for the view.[6]
- Second a copy of Zend Framework need to be developed. With the growth of complexity, additional code is required which will provide the functionality and that is relative small and focuses on the benefits of MVC system.[6] Regarding the process in more detail, the bootstrap file is initialization in one form or another.

Next it needs to be ensured the environment is correct and that there are no errors, followed by setting date and time for tracking functionality.[6] In order to set up date and time many procedures can be followed; for example the method data_default_timezone_set() can get called and Zend assumes that default directory will include the phd path.[6] The Zend Framework does not depend on any specific file, but helper classes are helpful in this case. Following are some examples:

- Zend_Loader::loadClass() the main purpose here is to correct file for the supplied class name.
- Following this the underscores are converted into directory-specific structures.[6] As a result, the code lines Zend_Loader::loadClass('Zend_Controller_Front'); and include_once 'Zend/Controller/Front.php'; show similar results.
- Zend_Debug::dump() functions in terms of debugging information and is focused on formatted var_dump() output.[6] Finally the bootstrap runs the front controller and initializes it. The design pattern used by Zend_Controller_Front is the Singleton design and getInstance() is used to get the single instance.[6]

12. Current Development

Zend Framework 3.0 was released on June 28, 2016. It includes new components like a JSON RPC server, a XML to JSON converter, PSR-7 functionality, and compatibility with PHP 7. Zend Framework 3.0 runs up to 4 times faster than Zend Framework 2, and the packages have been decoupled to allow for greater reuse.[25] The contributors of Zend Framework are actively encouraging the use of Zend Framework version 3.x. The stated end of life for Zend Framework 1 is 2016-09-28, and for Zend Framework 2 is 2018-03-31. The first development release of Zend Framework 2.0 was released on August 6, 2010.[26] Changes made in this release were the removal of `require_once` statements, migration to PHP 5.3 namespaces, a refactored test suite, a rewritten `Zend\Session`, and the addition of the new `Zend\Stdlib`. The second development release was on November 3, 2010.[27] The first stable release of Zend Framework 2.0 was released 5 September 2012.[28]

References

1. Gutmans, Andi (2005-10-27). "Zend Framework (post is too long so make sure to grab coffee)". Andi on Web & IT. http://andigutmans.blogspot.com/2005_10_01_archive.html. Retrieved 2009-02-11.

2. "Contributor Guide (ZF v1)". http://framework.zend.com/participate/contributor-guide-v1.
3. "Documentation for the ZF components". https://docs.zendframework.com/.
4. Pope, Keith. Zend Framework 1.8 Web Application Development (1). Olton, GB: Packt Publishing, 2009. ProQuest ebrary. Web. 13 February 2017.
5. Padilla, A. (2009). Beginning Zend Framework. Apress.
6. Allen, R., Lo, N., & Brown, S. (2009). Zend framework in action. Manning.
7. Company, Zend, a Rogue Wave. "Zend Framework - Issue". https://framework.zend.com/manual/2.3/en/user-guide/forms-and-actions.html.
8. Company, Zend, a Rogue Wave. "Zend Framework - Issue". https://framework.zend.com/manual/1.12/en/zend.controller.router.html.
9. Company, Zend, a Rogue Wave. "Zend Framework - Issue". https://framework.zend.com/manual/2.4/en/modules/zend.view.quick-start.html.
10. "History of PHP and related projects". The PHP Group. http://www.php.net/history. Retrieved 2009-02-11.
11. LaMonica, Martin (2005-02-25). "IBM backs open-source Web software". cnet.com. http://news.cnet.com/IBM-backs-open-source-Web-software/2100-7344_3-5589559.html?tag=nw.14. Retrieved 2009-02-11.
12. Kernel, Sean Michael (2006-12-20). "Google Data Joins PHP Zend Framework". internetnews.com. http://www.internetnews.com/dev-news/article.php/3650066. Retrieved 2009-02-11.
13. Krill, Paul (2006-10-31). "Microsoft, Zend boost PHP for Windows". infoworld.com. http://www.infoworld.com/article/06/10/31/HNzenphp_1.html. Retrieved 2009-02-11.
14. Potter, Mike (2014-05-21). "Adobe Contributing AMF Support to Zend Framework". The Official Flex Team Blog. http://blogs.adobe.com/flex/archives/2008/07/adobe_contributing_amf_support.html. Retrieved 2009-02-11.
15. "StrikeIron Featured Partners". http://www.strikeiron.com/partners/featured_partners.aspx. Retrieved 2009-02-11.
16. "About Zend Framework". http://framework.zend.com/about/overview. Retrieved 2009-02-11.
17. Why to Use Zend Framework? By SuntecOSS, Retrieved, April 21st, 2016 http://www.suntecoss.com/blog/why-to-use-zend-framework/
18. Zend's Apigility, an Open Source API Builder for Developing Quality APIs By SuntecOSS, Retrieved, May 19th, 2016 http://www.suntecoss.com/blog/zend-apigility-an-open-source-api-builder-for-developing-quality-apis/
19. "Zend site". http://www.zend.com/products/server.
20. "Download Zend Studio - IDE, PHP profiler, mobile, unit testing & more". http://www.zend.com/en/products/studio/features#ZFI.
21. "Zend Framework Contributor Guide". July 1, 2006. http://framework.zend.com/wiki/display/ZFDEV/Zend+Framework+Contributor+Guide.
22. "Simple Cloud API Press Release". Archived from the original on December 1, 2009. https://web.archive.org/web/20091201014623/http://www.zend.com/en/company/news/press/zend-teams-with-ibm-microsoft-rackspace-and-other-cloud-leaders-on-open-source-initiative-to-drive-cloud-application-development.
23. "Zend Framework website". http://framework.zend.com/.
24. simplecloud.org http://www.simplecloud.org
25. zendframework (2016-06-28). "Zend Framework 3 Released!". https://framework.zend.com/blog/2016-06-28-zend-framework-3.html.
26. "Zend Framework 2.0.0dev1". 2010-08-06. http://devzone.zend.com/article/12385-First-Development-Milestone-of-ZF-2.0-Released.
27. "Zend Framework 2.0.0dev2". 2011-11-03. http://framework.zend.com/announcements/2010-11-03-zf2dev2. Retrieved 2011-03-18.
28. "Zend Framework 2.0.0 STABLE Released! - Zend Framework - Zend Framework". Framework.zend.com. September 5, 2012. http://framework.zend.com/blog/zend-framework-2-0-0-stable-released.html.
