Comparing Zend Framework and Yii MVC Frameworks for PHP

Overview and Management Summary:

This document is based on a review of the documentation for the Zend Framework and Yii object oriented Model View Controller platforms used for development of PHP web applications. This document provides point by point comparisons of the platforms for areas of interest which may be relevant to our web development. Links to relevant documentation are also provided. In the section below links are also provided to third party reviews comparing the two platforms.

Though in many ways the platforms are similar it is this writer’s opinion that Yii may have certain key advantages for us. In particular its better documentation, ease of use and lower complexity will probably lead to quicker learning curves and faster development as well as less frustration and pulling out of hair by some already nearly hairless developers.

Outside opinions (mostly pro Yii):

<http://www.larryullman.com/2011/06/01/yii-vs-zend-vs-code-igniter-compared/>

<http://www.sheldmandu.com/php/php-mvc-frameworks/yii-vs-zend-vs-code-igniter-compared>

<http://brixican.blogspot.com/> Scroll to bottom.

<http://www.backwardcompatible.net/post/8961623281/7-reasons-why-yii-framework-is-better-than-codeigniter>

Comparisons by Issue:

Installation:

Bot Yii and Zzend are installed by placing the framework directories where the application can access them. For Zend if the framework was not installed as part of Zend server the php.ini include\_path setting may need to be modified to point to this location. For YII th yiic command line tool used to create an application appears to configure the app to point to the correct location based on where yiic is installed.

Zend: <http://framework.zend.com/manual/en/learning.quickstart.create-project.html>

Yii: <http://www.yiiframework.com/doc/guide/1.1/en/quickstart.installation>

Configuration:

Both Zend and Yii use configuration files in the application’s directories for setting deployment specific parameters such as database settings. By default Zend Uses hierarchically scoped names value pairs in a text file (application.ini) which can be accessed in code as a php hierarchy of arrays. YII stores config settings directly in a php file (config.php) as name value pairs in a hierarchy of nested arrays. (Zend also has classes to support this but the default code would need t obe modified ot use them.) Zend has separate scoped sections in the config file for production, test, development etc. The appropriate selection is selected based on an environment variable “APPLICATION\_ENV” (set in their demo in the .htaccess file.) Yii by default reads a single set of configuration variables but since they are in a php file different settings or files can be selected programmatically based on environment variables.

Zend: <http://framework.zend.com/manual/en/zend.application.quick-start.html> <http://framework.zend.com/manual/en/zend.config.introduction.html>

Yii: <http://www.hollowdevelopers.com/2011/05/14/yii-framework-separate-configurations-for-different-environments/>   
<http://www.yiiframework.com/wiki/289/use-application-on-production-development-environment-without-making-changes/>

Project Creation and Code Generation

Both Yii and Zend provide command line tools for creating and configuring applications. (zf and yiic.) The zend tool (zf) requires knowledge of specific command line arguments in order to use it for various tasks such as creating data model code. Yii provides a separate web based GUI (Gii) which can be used for such tasks.

Yii: <http://www.yiiframework.com/doc/guide/1.1/en/quickstart.first-app>   
 <http://www.yiiframework.com/doc/guide/1.1/en/topics.gii>

Zend: <http://framework.zend.com/manual/en/learning.quickstart.create-project.html>

Application Organization

Both Zend and YII use similar directory structure for deploying web applications. The application has folders for the application, assets, third party libraries, modules and the public directory where the index.php and .htaccess files can be, as well as folders or sub folders for models views and controllers.

Yii: <http://www.yiiframework.com/doc/guide/1.1/en/quickstart.first-app> <http://www.yiiframework.com/doc/guide/1.1/en/basics.convention#directory>

Zend: <http://framework.zend.com/manual/en/learning.quickstart.create-project.html>

Architecture:

Both Zend and Yii follow similar over all approaches to implementing MVC (Model View Controller) architecture.

Yii: <http://www.yiiframework.com/doc/guide/1.1/en/basics.mvc>

Zend: <http://framework.zend.com/manual/en/learning.quickstart.intro.html>

Database Models:

Both Zend and Yii use config file settings along with PHP Data Objects(PDO) to set up the database connection. Yii creates a Database Access Object (DAO) while Zend creates a db-adapter based on the config settings.

In Yii database models are built using object relational mapping by extending an active record super class for each database table. Each instance represents a record, table level access and queries are handled by class level methods. A class for a single table can be defined by creating a sub class and defining the table name. The superclass accesses the database schema to provide accessors and attributes. Mutli-table relationships are handled by creating classes for each table but if the foreign key dependencies are defined in the schema then each table class will have accessors to access dependent records in the other tables or to create joins. More complex models can be built by extending these classes or aggregating them in a combined model class and by using the Query Builder.

<http://www.yiiframework.com/doc/guide/1.1/en/database.ar> <http://www.yiiframework.com/doc/guide/1.1/en/database.arr> <http://www.yiiframework.com/doc/guide/1.1/en/database.query-builder>

Zend uses the Table Data Gateway pattern to abstract database tables. One instance represents the table . Methods access records. Even for single table models Zend uses a separate data mapper class to abstract actions on the table as well as a separate model class. The controller manages both the mapper and model classes. Zend provides automatic code generation for table, mapper and model classes based on the database schema. These classes can be extended or modified for more complex requirements.

<http://framework.zend.com/manual/en/learning.quickstart.create-model.html> <http://framework.zend.com/manual/en/zend.db.table.html> <http://www.martinfowler.com/eaaCatalog/tableDataGateway.html>   
<http://survivethedeepend.com/zendframeworkbook/en/1.0/implementing.the.domain.model.entries.and.authors>

Forms and Views

Both Zend and YII use a combination of hard coded html and programmatic adding of elements to create views and forms. They both tend to differentiate between forms and views even though with Ajax any view can act as a form. They both tend to use the programmatic creation of elements only for the forms and then include the forms in views. Both Zend and Yii support using a common layout view as a container for other views in order to use common code and provide a common look and feel.

Zend: <http://framework.zend.com/manual/en/learning.quickstart.create-form.html>  
<http://framework.zend.com/manual/en/zend.view.introduction.html>   
<http://survivethedeepend.com/zendframeworkbook/en/1.0/setting.the.design.with.zend.view.zend.layout.html.5.and.yahoo.user.interface.library>

Yii: <http://www.yiiframework.com/doc/guide/1.1/en/basics.view>   
 <http://www.yiiframework.com/doc/guide/1.1/en/form.overview>

Style and Presentation:

Both Yi and Zend appear to support the use and loading of separate css and js files or segments for different components. They also both can use classes assigned to elements in the html code of views to associate those elements with specific styles.

Zend uses the decorator pattern to modify the appearance of programmatically created form elements Zend also uses view helpers to produce reusable formatted components.

Yii comes with many out of the box components and widgets that include css. These are defined in such classes as CMenu and CHtml.

Zend: <http://framework.zend.com/manual/en/learning.form.decorators.html>  
<http://devzone.zend.com/1651/managing-css-and-javascript-files-within-a-zend-framework-app/>  
<http://framework.zend.com/manual/en/zend.view.helpers.html>

Yii: <http://www.yiiframework.com/wiki/211/creating-a-css-driven-drop-down-menu-using-cmenu/>   
<http://www.yiiframework.com/doc/api/1.1/CHtml>

jQuery and Ajax Integration:

Both Zend and Yii support the integration of jQuery and Ajax functionality into GUI (View) components (Widgets.) Both foundations provide a set of existing such active components. In Yii existing widgets and components can be used with a function call at one place in the view or form Zend is similar however additional step(s) appear to be required to load and instantiate the appropriate view helper. Below are some references:

Zend: <http://framework.zend.com/manual/en/zendx.jquery.view.html>

Yii: <http://www.yiiframework.com/wiki/49/update-content-in-ajax-with-partialrender/> ,  
<http://www.yiiframework.com/doc/api/1.1/CHtml>  
<http://usingjquery.com/2010/10/using-jquery-with-the-yii-framework/>   
<http://learnyii.blogspot.com/2010/12/yii-using-jquery-ui-dialog-boxes.html>   
<http://www.yiiframework.com/doc/guide/1.1/en/extension.create#widget>  
<http://www.yiiframework.com/doc/guide/1.1/en/form.view>

Validation:

Both Zend and YII provide many standard validators which can be used to check user input. With both frameworks you can also create your own custom validator classes. In Zend validators are associates with form elements by calling $element->addValidator(). In Zend models access the form to validate data. In Yii validation is handled in the model by filling out the rules array which associates validation rules with the attributes of the model.

Zend: <http://framework.zend.com/manual/en/zend.validate.html>   
<http://framework.zend.com/manual/en/zend.validate.set.html>   
<http://framework.zend.com/manual/en/zend.form.elements.html>

Yii: <http://www.yiiframework.com/wiki/56/>

Error Handling:

Zend and Yii have similar approaches to error handling. Unless configured otherwise Zend expects an error controller and associated views to be in place to handle exceptions. In YII the application component of Class CErrorHandler handles uncaught exception and looks for a properly named and located view to display information about the error. Both Zend and Yii can be configured for development to display the complete exception stack rather than the error view.

Zend: <http://survivethedeepend.com/zendframeworkbook/en/1.0/handling.application.errors.gracefully>

Yii: <http://www.yiiframework.com/doc/guide/1.1/en/topics.error>   
 <http://www.yiiframework.com/doc/api/1.1/CErrorHandler>

Auto Loading

Both Zend and Yii have auto loading mechanisms to load php files as needed so you don’t need “require\_once” statements everywhere and you do not need to load files unless they are needed.

Zend: <http://framework.zend.com/manual/en/learning.autoloading.html>

Yii: <http://www.yiiframework.com/wiki/165/understanding-autoloading-helper-classes-and-helper-functions/>

Modules

Both Zend and Yii support use of modules to break an application into parts. Each module is like a mini-application and has its own models, views and controllers. Modules facilitate project organization as well as code reuse.

Zend: <http://framework.zend.com/manual/en/zend.controller.modular.html>

Yii: <http://www.yiiframework.com/doc/guide/1.1/en/basics.module>

OOP-ness

Both Zend and Yii are well organized object oriented frameworks. Zend seems more severely object-oriented in its use of more layers of abstraction and more complex patterns. Though this may allow slightly better encapsulation and code reuse the benefits are offset by greater complexity, code bloat and a longer learning curve.

Code reuse:

Both frameworks support code reuse through inheritance, helper classes, Components, Widgets, Libraries etc.

Complexity of code structure, configuration

Zend applications seem to end up with a more complex code structure and a more complex configuration set up.

Accessibility:

Initial search does not show that either framework directly addresses accessibility guidelines. This would probably still be our responsibility as programmers and designers.

Security:

Both platforms support the development of secure applications and provide standard classes and functions to prevent cross site scripting and sql injection.

Yii: <http://www.yiiframework.com/doc/guide/1.1/en/topics.security>

Zend: <http://framework.zend.com/security>   
 <http://static.zend.com/topics/Webinar-Zend-Secure-Application-Development-with-the-Zend-Framework.pdf>

Documentation:

Both platforms have fairly extensive online documentation as well as third party tutorials and books available. In general I have found the documentation for Yi better organized, easier to read and easier to follow and understand. This tends to follow from the lower complexity of some of the patterns as well as from writing style.

Zend: <http://survivethedeepend.com/zendframeworkbook/en/1.0>  
<http://framework.zend.com/manual/en/>

Yii: <http://www.yiiframework.com/doc/>   
<http://www.yiiframework.com/doc/guide/>   
<http://www.yiiframework.com/doc/api/>

Programming Style and Conventions:

Both platforms have documented coding style and naming convention standards.

Yii: <http://www.yiiframework.com/doc/guide/1.1/en/basics.convention>

Zend: <http://framework.zend.com/manual/en/coding-standard.html>

Naming rules and auto-magic functionality

Both platforms require the use of the use of certain naming and directory organization standards in order for certain built in functionality to work. Common class naming is used to associate models views and controllers. Controller class and action method naming needs to match URL strings in order for the correct actions to be called. For example with Zend default behavior will render the view with a matching name for a controller without explicitly being called.

Maintainability

Both platforms by encouraging a standard architecture, code reuse and common coding and documentation standards will result in reasonably maintainable code. Yii has the additional advantage of a faster learning curve and less complexity leading to faster maintenance.

Learning curve:

My initial reviews of the documentation indicate that Yii has a significantly faster learning curve (on the order of 2-3 times.)

Portability, Deployment, Migration, Version control:

Both platforms have similar approaches. Neither one sets you ups for a particular version control system. Both include all assets and third party code in the deployment directory hierarchy for ease of deployment. Yii also has a class that provides functionality for database migration when a code change requires it.

Yii: <http://www.yiiframework.com/doc/guide/1.1/en/database.migration>

Testing:

Both platforms provide support for unit testing and functional testing:

Yii: <http://www.yiiframework.com/doc/guide/1.1/en/test.overview>

Zend: <http://framework.zend.com/manual/en/zend.test.html>

Mobile:

Neither platform appears to have specific classes to address mobile apps but both have third party extensions available for mobile device detection:

Zend: <http://www.handsetdetection.com/blog/mobile-detection-with-the-zend-framework/>

Yii: <http://www.yiiframework.com/extensions/?tag=detect+mobile+browser>

The Future:

Hopefully both platforms will be around for a while and growing to address new needs as they evolve. Zend has the advantage of the backing of a large company and its resources. Yii has the advantage of growing popularity and a strong developer community.