基于 PHP 的 M-V-C 开源框架调研报告

南京安元科技有限公司
黄思夏

March 12, 2010

调研背景

PHP 其本质是一种解释执行的脚本语言。因快速开发、自由度大、便于扩展而备受关注。经过 5 个主要版本的发展，PHP 已从主要服务于个人网站和小规模项目的定位发展成可靠而广泛的 Web 服务器脚本。

PHP 设计阶段是面向过程的，但越来越多面向对象的特性被整合进来。Web 2.0 时代的来临也对 Web 开发工具提出了更高的要求。更快速的开发模式，管理全局项目的能力，更易于扩展的能力正考验着已有的开发工具。因此在传统软件开发领域就存在的 MVC 概念再次火热起来， ROR 框架应运而生、势头迅猛。因此，基于 PHP 的，MVC 框架也如雨后春笋，不断推陈出新。

在经历一段时间的竞争、分裂、整合和市场消化后，一些设计优秀的 PHP 框架（PHP Frameworks）生存了下来，并逐步成长和完善。越来越多的项目也选择搭建在已有的成熟框架上而免去从头开发的繁琐，而且更能整体把握项目的性能和代码可靠度，同时也大大加快了项目的开发进度。

面对种类繁多的 PHP 框架，如何选择适合目标项目的，效率高而稳定的，有前景的框架，成为项目伊始必须考虑的重要问题之一。

主流的 PHP 框架

主流并可以运用在实际项目的成熟框架有：CodeIgniter、Kohana、Yii (Yii Web Programming Framework)、Zend (Zend Framework)、Symfony 以及 CakePHP 等等。综合部署环境、开发效率、中文支持、性能、承载能力、灵活性、扩展性、框架前景、学习曲线等多方面考虑，本文抽出综合素质比较好的四个框架进行分析，分别是：

CodeIgniter

官方网站：http://framework.zend.com/

Kohana

官方网站：http://www.Kohanaphp.com/

Yii

官方网站：http://www.yiiframework.com/

Zend

官方网站：http://framework.zend.com/

部署环境

目前 PHP 版本已经升级到 5.x 了，但是有部分旧的 Web 服务器依然运行着 PHP 4.x。PHP 5.x 带来很多性能和安全性的升级，例如 MySQLi 的引入大大改善 PHP 访问 MySQL 的性能。因此，只兼容 PHP 5.x 的框架相对同时兼容 PHP 4.x 和 PHP 5.x 的架构，理论上在执行效率方面会有优势。但是框架的实际执行效率也同时受到其他各方面的制约，因此并不能草率下判断。

下面列举一下各个 PHP 框架要求的部署环境：

CodeIgniter

PHP 版本：4.3.2 或更高
数据库接口：MySQL(4.1+), MySQLi, MS SQL, Postgres, Oracle, SQLite and ODBC

Kohana

PHP 版本：5.2.3 或更高
数据库接口：MySQL, MySQLi, SQLite and PostgreSQL
其他条件：Web 服务器必须支持 Unicode

Yii

PHP 版本：5.2.3 或更高
数据库接口：MySQL, MySQLi, SQLite and PostgreSQL

Zend

PHP 版本：5.2.4 或更高
数据库接口：MySQL(4.1+), MySQLi, MS SQL, Postgres, Oracle, SQLite and ODBC

开发效率

为实际体验框架的开发效率以及性能表现，我分别用四个 PHP 框架编程实现了同一个简单的通讯录(Addressbook)应用。

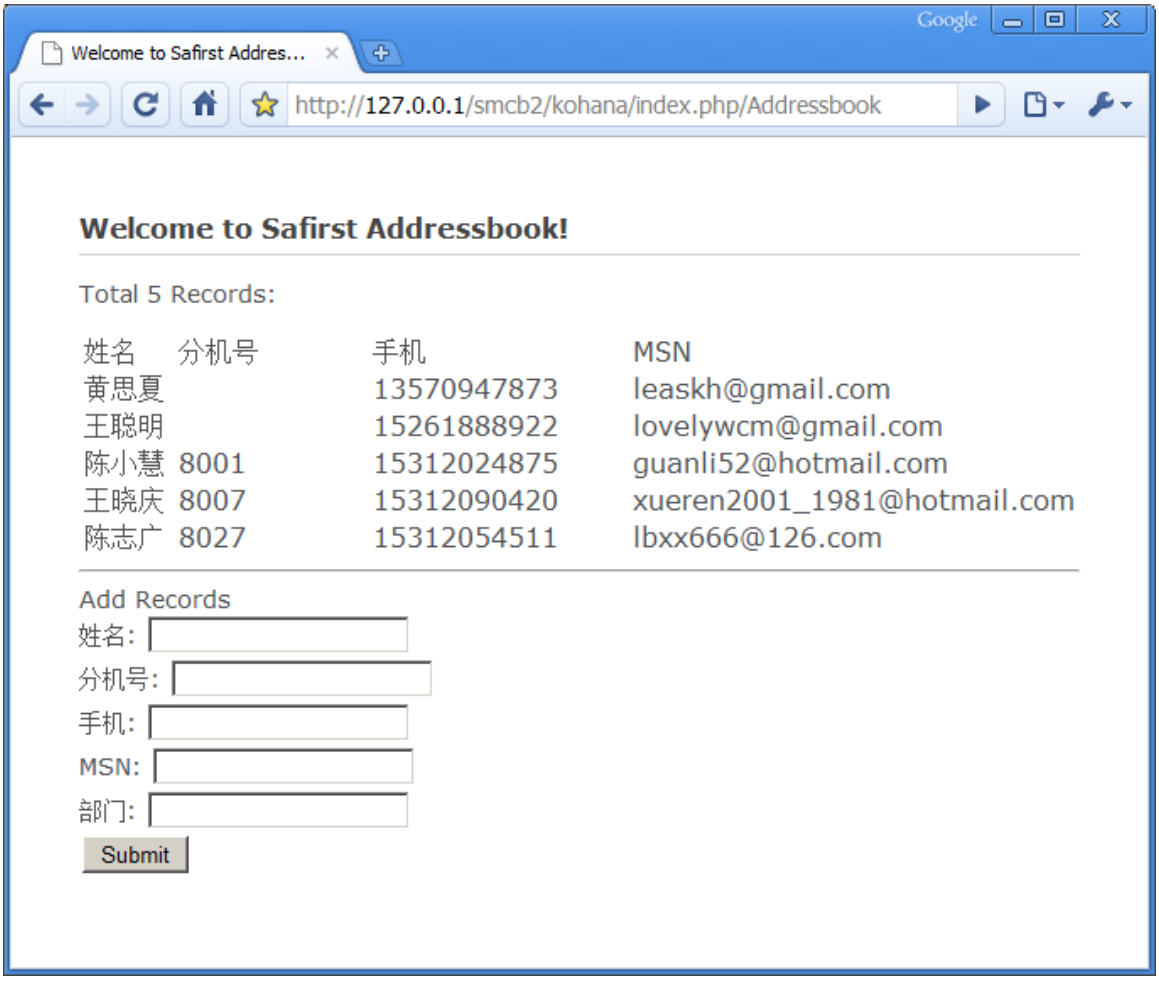
实验环境

Server:
Windows 7 + Java 1.6.0_18 + Apache 2.2.14 + PHP 5.3.2 + MySQL 5.0.7

Version Control:
TortoiseHg 0.9.3 with Mercurial-1.4.3, Python-2.5.4, PyGTK-2.12.1, GTK-2.16.1

IDE（IDE 的调研详情请参见：《附录 1》）：
Eclipse for PHP Developers 1602 + NetBeans PHP IDE 6.9m1 with xdebug 2.1.0_rc1

下图是程序执行效果：



PS：由于搭建的时候每个架构的 View 层都使用了相同的 HTML + CSS，所以四个程序的界面几乎完全一致，这里就不一一截图了。

CodeIgniter

从最熟悉的入手，我首先尝试的是 CodeIgniter 架构。CodeIgniter 框架部署好以后，设置好数据库连接文件和 URL 路由，只需要新建 Addressbook 控制器，写入代码：

```
<?php
class Addressbook extends Controller
{
	function Addressbook()
	{
	    parent::Controller();
	    $this->load->helper('url');
	    $this->load->scaffolding('main_db');
	}
	function index()
	{
	    $data['addresult']=$this->db->get('main_db');
	    $this->load->view('addressbook_show',$data);
	}
}
?>
```

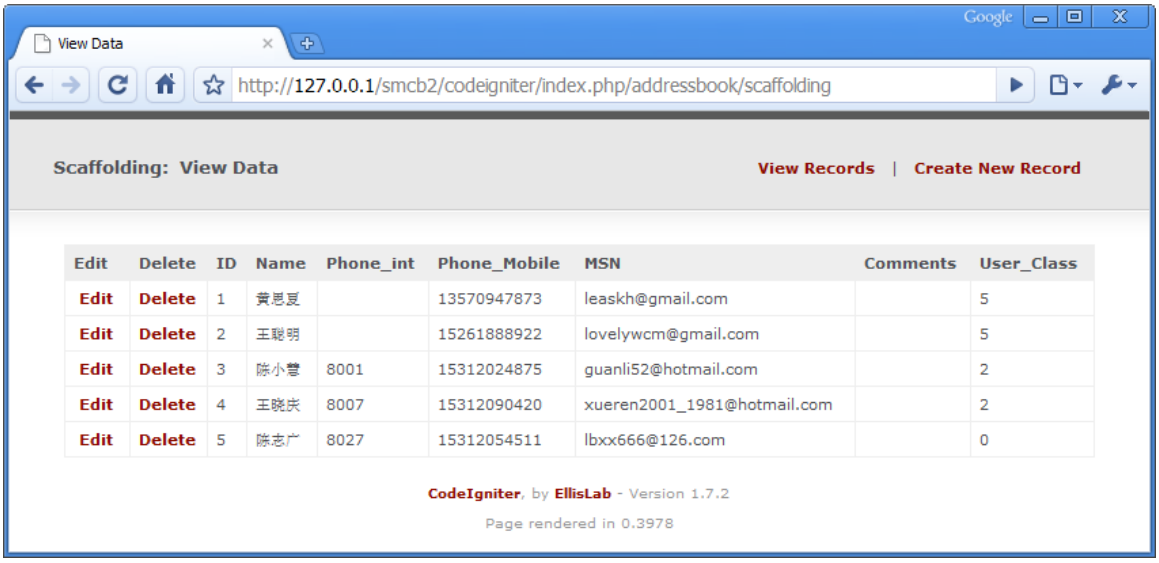
然后就可以从视图 address_show 中访问控制器传递的变量，程序就执行起来了。*PS：视图中的 HTML 代码及简单 PHP 变量访问这里就省略了。*

CodeIgniter 的层次清晰，调试便利，大大有利于实际开发。如果意外遇到代码错误，CodeIgniter 能够清晰准确地指出错误所在，对开发者十分友好，如：

```
An Error Was Encountered
Unable to load the requested file: addressbook_show_err.php
```

还有一点值得提出的是 CodeIgniter 框架内置一个很方便的数据库搭建辅助工具 scaffolding(脚手架)。透过 scaffolding，在程序的开发阶段能够很方便地添加，删除和修改数据库条目、生成调试用的数据等，而不需要来换切换于 IDE 和 phpMyAdmin 或者命令行 MySQL 之间，有利于提高开发效率。

下图为 scaffolding 的截图：



Kohana

由于 CodeIgniter 源于 Kohana，由开源社区驱动发展而来，在路由设置，数据库访问等细节上都与 CodeIgniter 如出一辙。所以 CodeIgniter 中完成的代码简单迁移到 Kohana 然后稍作修改，就能运行良好。主要的修改是在 Kohana 中，类是自动加载的，而不需要像在 CodeIgniter 中那样，通过 load 语句实现：

```
......
$this->load->helper('url');
$this->load->scaffolding('main_db');
......
$this->load->view('addressbook_show',$data);
......
```

在 Kohana 中，类可以直接实例化，实现真正的自动加载并立刻投入使用：

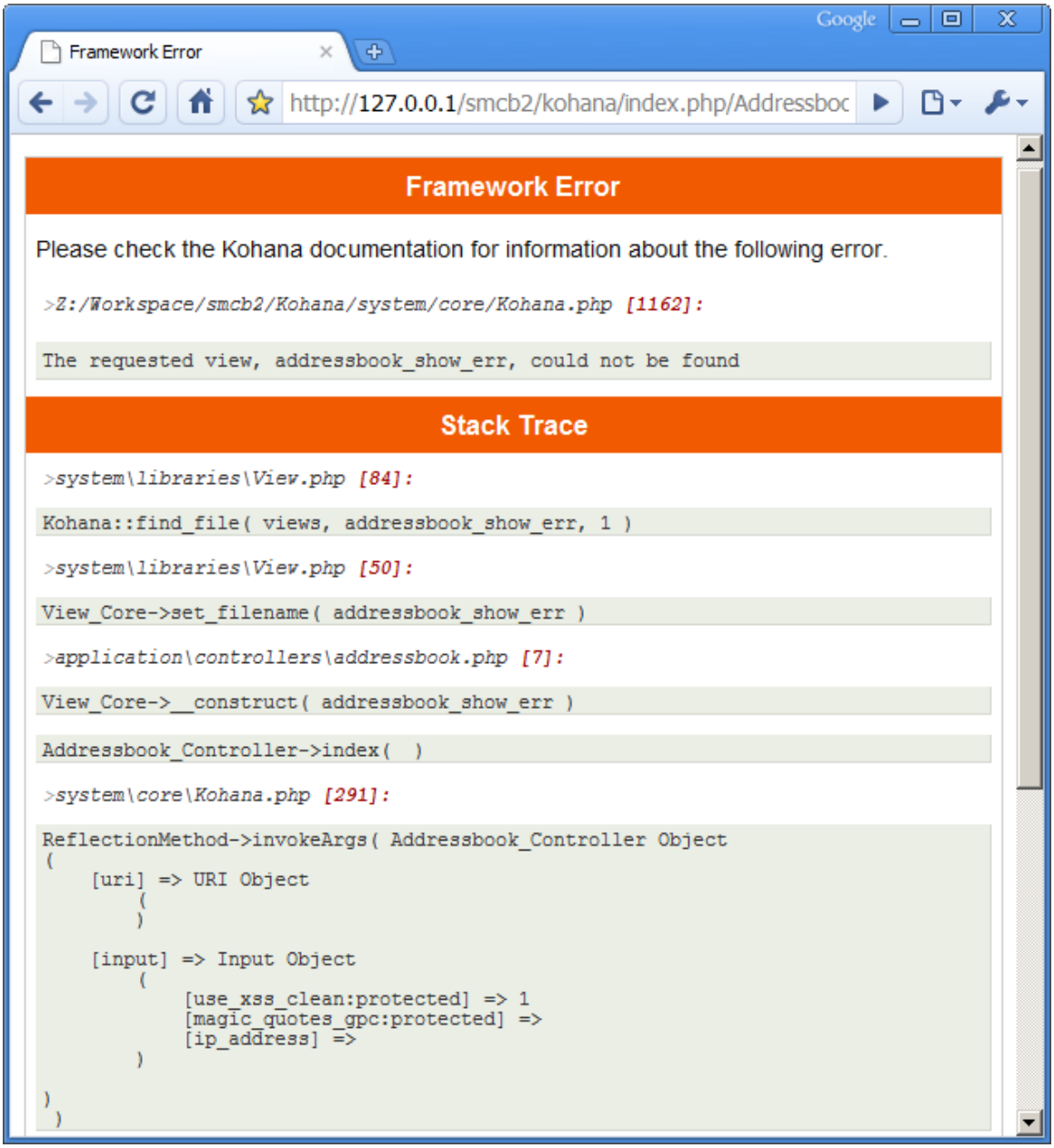
```
......
$view = new View('addressbook_show');
......
$view->render(TRUE);
......
```

Kohana 有一个特性是类的名称相对不自由，例如继承自“Controller”的类，声明的时候就必须在类名后面加“_Controller”后缀,如：

```
.....
class Addressbook_Controller extends Controller
{
.....
}
.....
```

这一点不好适应，感觉过于繁琐。好处是：无命名空间的冲突。 类均添加了如“_Controller” 之类的后缀，从而使得用户的控制器和数据模型可被同时同地装载。

然而让我印象深刻的是，Kohana 的调试功能十分强大，错误输出信息很丰富：



Kohana 不单单输出框架外部用户代码中发现的错误，而且通过 Stack Trace(堆栈跟踪)输出框架内部的运行状况，有利于确认问题是出在用户代码还是框架本身，对解决问题帮助很大。

Yii

Yii 是海外华人主导开发的，以谐音“易 -> Yii”得名。Yii 的结构简单，在项目最初期可以通过辅助脚本帮助建立应用的实际框架，复制必要的文件到实际项目文件夹。

在 Yii 的文件系统结构中，架构被单独放在一个独立的 Frameworks 文件夹，然而，应用需要的用户代码则放在其他构建的文件夹中，这一点和上两个框架稍有不同。Yii，通过应用目录的 index.php 中写入代码加在整个架构：

```
<?php
// include Yii bootstrap file
require_once(dirname(__FILE__).'/../framework/yii.php');
// create a Web application instance and run
Yii::createWebApplication()->run();
?>
```

这一点相对前面两个框架来说，稍微复杂。Yii 架构的内核比较小，很多功能都是通过官方和非官方插件的形式实现。配置的自由度比较大。Yii 框架的控制层通过 render 实现加载，传递数据并渲染是图层，十分便利的。如：

```
$this->render('addshow', array(
        'rsadds'=>$rows
));
```

Yii 的视图层也细分为两个层次，分别是 site 和 layouts，其中 site 层管理的是视图输出的内容，layouts 管理的是视图输出的形式（格式）。控制器通过 rander 函数读取 site 层的内容，结合控制器传递的变量，渲染成为实体内容，存放在$content 变量中。最后还需要通过 layouts 层通过<?php echo $content;?>实现真正的输出。这一点乍看是复杂化了视图层的结构，但是同时为代码重用创造了更大的可能性。例如一个 Blog（博客）使用 Yii 搭建，那么其他框架就很可能需要加载不同的视图层实现输出 HTML 和 Feed，但是，在 Yii 的体系中，只需要新建一个 Layout 就可以实现两种输出了，这一点在应用程序需要开发 API 实现扩展多种输出的时候，能提供很大的灵活性。

Yii 比较致命的缺点是调试十分不便利，用 Yii 架构编写程序的时候，如果发生编码错误，Yii 绝大多数情况下不能正确输出错误所在，往往只是显示服务器错误，或者干脆显示一个空白页，没有任何提示。这一点很不友好，往往需要第三方的调试工具。

Zend

Zend 来自 Zend 公司，著名的 Zend PHP Engine 就是 Zend 公司出品的。应该说 Zend 是一个相对比较官方的 PHP 框架，而且 Zend 的开发社区事实上也是受到 Zend 资助的。Zend 也是众多架构中，最庞大的架构之一，Zend 目标明确：面向企业用户，满足大型企业网站的开发需求。

Zend 框架高度模块化，几乎每个细节都可以定制，自由度也比较大。Zend 辅助脚本帮助开发人员配置和部署项目。由于 Zend 的庞大，上手 Zend 是比较难的，同一应用程序用 Zend 实现的代码量也是最大的。

Zend 的代码编写要求很严格。Zend 有数十条编码规则（Code Rule）：http://framework.zend.com/manual/en/coding-standard.naming-conventions.html 规范了包括：PHP 标记、文件命名、代码缩进、类名、下横线的用法、甚至换行符的用法等等纷繁细节。这些规定虽然让代码更加严谨可靠，但是同时给开发人员带来诸多不便，严重影响了整体的开发效率，抹煞了 PHP 快速开发的优点。使得 Zend 搭建项目的耗时和使用 J2EE 相当。

Zend 的调试信息也不友好，输出的错误信息往往对解决问题完全没有帮助，也同样存在 Yii 的有错不报的现象。

中文支持

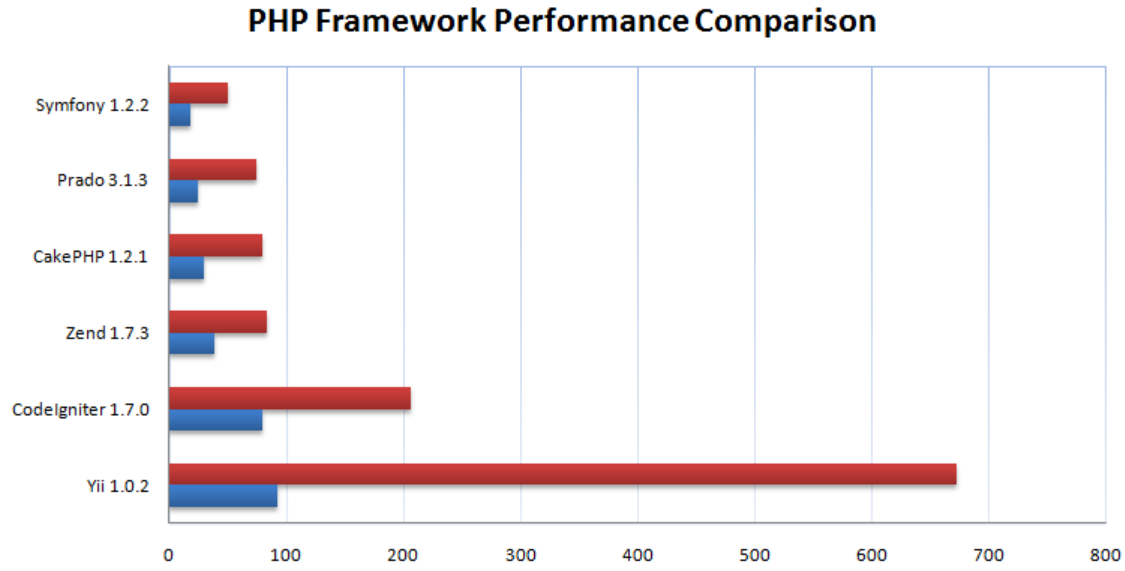
中文支持方面，CodeIgniter、Kohana、Yii 和 Zend 都可以支持中文。其中 Kohana 声称 100% Unicode 运行（因此部署环境需要 Server 支持 Unicode 环境），更能减少乱码产生。

实际开发中，MySQL 设置为 Unicode 模式，数据编码设置为 Unicode 模式的情况下。CodeIgniter、Kohana 和 Zend 都能顺利读写中英文字符，只有 Yii 出现中文乱码，需要独立设置 Yii 的 MySQL 连接对象 CDbConnection 的字符集属性'charset'=>'utf-8'然而这个属性只能在配置阶段设置，并不能在“运行时（Runtime）”修改，带来一定的不便。

性能分析

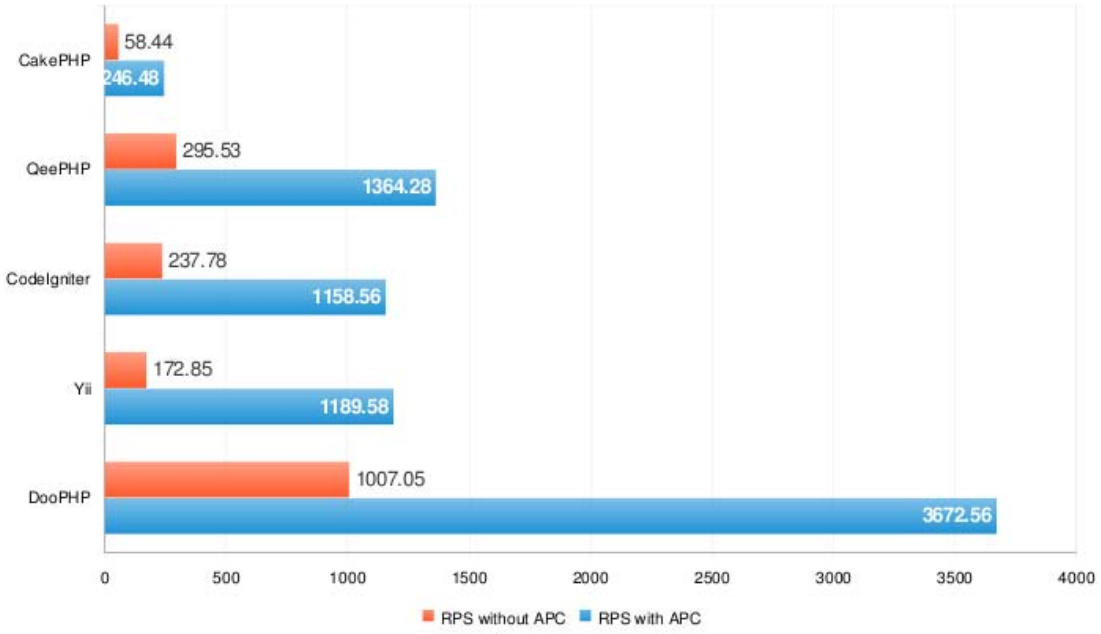
由于测试项目比较小，程序的反应速度都是相当快的。只有某一些情况下，能感受到 Zend 操作数据库的动作有所延迟。Zend 的性能相对其余三个来说，并不太理想，这个可能是因为 Zend 架构庞大，资源调用比较频繁的缘故。

Yii 官方网站中给出一组评测数据：



	Yii 1.0.2	CodeIgniter 1.7.0	Zend 1.7.3	CakePHP 1.2.1	Prado 3.1.3	Symfony 1.2.2
RPS with APC	673	206	83	79	75	50
RPS without APC	93	79	38	29	24	18

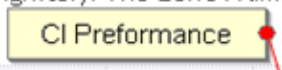
数据显示，Yii 的确性能卓越（不排除广告成分），CodeIgniter 的性能也是比较理想的。另外 Doophp 上也有一组数据可供参考： http://www.doophp.com/blog/page/5/



最后还有一组由 avnetlabs 发布的评测数据，比较客观和权威：
http://avnetlabs.com/php/php-framework-comparison-benchmarks/

- **No PHP code cache**

All frameworks used an ORM (of sorts, in the case of codeigniter). The Zend Framework used Zend_DB_Table and CodeIgniter used ActiveRecord.



	Run 1	Run 2	Run 3	Run 4	Average
Baseline HTML	1327.5	1326.5	1328.6	1329.1	**1327.9**
Baseline PHP	331.6	332.1	331.4	332.0	**331.8**
CakePHP	3.6	3.7	3.8	3.5	**3.7**
CodeIgniter	21.5	21.2	21.7	21.7	**21.5**
Zend Framework	9.3	9.1	9.2	9.3	**9.2**

这组数据同样也支持 CodeIgniter 的性能比较好。

承载能力

我认为框架的承载能力可以从两个角度谈，一个是性能层面的承载能力，一个是纯应用层面的承载能力。

承载能力其实是相对的，执行速度慢的框架由于需要占用代码文件、脚本文件、页面资源的时间比较长，同时需要占用数据库的会话时间也比较长，所以会一定程度制约整个服务器所能承载的极限；相反，较快地释放这些资源将有助于服务器资源的回收和循环，提供比较高的“并发”。Zend 虽然能提供企业级别的服务器编程架构，这个架构在大型的商业网站实践上是可行的，因为大型网站一般配备比较充足的运算资源。然而在相同级别的服务器上，运行简单框架带来的效率提升在所带来的并发访问流畅度的提升，也是明显的。

其次是应用层面的承载整理，如果忽略掉性能的考虑，的确 Zend 在应用层面的考虑是最充分的，在开发时间充足，资源充裕的情况下，使用 Zeng 能够让项目的功能有很大的延展可能性。结构完整的 Model 层，也是其他三个架构所不完全具备的。Zend 的 Model 层抽象化管理数据的能力最为强大。如果项目的数据结构比较复杂，那么 Zend 也是不二之选。当然，那是当项目很大的情况下，当项目规模中等，或者一般的大型项目，相信其他几个架构也可以承担。

灵活性和扩展性

CodeIgniter 的插件规模一般，CodeIgniter 提供官方扩展 API，便于开发。

Kohana 继承了 CodeIgniter 的大体框架，并提供比 CodeIgniter 更加强大的和灵活的插件接口，由于 Kohana 开发社区的活跃，Kohana 除了官方提供一些很有用的扩展外，第三方的扩展也十分繁荣。

Yii 也提供很完整的扩展 API，Widget、Action、Filter、Controller、Validator、Module 甚至 Console Command 都可以通过官方的 Api 进行扩展，但是 Yii 的社区相对来说比较小，实用化的扩展并不多。

Zend 框架本身可配置性很大，虽然官方也提供扩展组件和扩展 API，但是事实上就算不通过官方的接口，扩展和修改 Zend 都很灵活，只要不触及内核部分，Zend 的内核部分很严谨，缺乏经验的情况下很容易造成错误，而且 Zend 的调试功能做得不好，所以不容易发现修改过程中哪一部分出现了问题。

框架前景

CodeIgniter 由 ellislab（http://ellislab.com/）开发，该公司的主要架构是 ExpressionEngine（http://expressionengine.com/）一个纯商业运作的框架。CodeIgniter 也有开源社区，但是社区并不太活跃，主要的维护也是靠 ellislab 完成，ellislab 对社区的反馈也比较滞后，因此不少社区用户反应 CodeIgniter 的更新太慢，Bug 修复往往需要等上几个月。但是正是由于有商业公司的资助，CodeIgniter 的发展虽然慢，但是稳步向前，发展到目前已经很成熟，可以用于实际项目了。CodeIgniter 以后的前景很大程度依赖于 ellislab 公司的商业运作状况。

Kohana 的诞生自 CodeIgniter 社区，也正是因为不满 CodeIgniter 的发展，CodeIgniter 社区的一部分人组织起来开发了 Kohana 1.0。Kohana 2.0 以后的版本很大程度已经脱离了 CodeIgniter 的代码，改写或者重构了。Kohana 因此可以被视为一个新的框架，这个框架完全由社区驱动，目前发展迅速。也因为这样，Kohana 的版本跳跃度很大，很多时候一个 API 在一个次版本的升级中都会被修改，代码的结构调整也是比较频繁的。如果作为实际项目的使用，可能造成日后升级项目的时候不能平滑过渡。客观说，Kohana 面向大型项目，还欠缺成熟。

Yii 从一开始就是由开源社区主导开发的，Yii 核心团段也比较小，只有不到 10 个人。Yii 由于结构简单，而性能卓越。但是随着代码量不断的增加，功能和性能之间的平衡随时可能变动。Yii 社区目前来说也相对较小，但是 Yii 社区很活跃，基本上一天内的 Commit 就有几十到上百个，发展很迅速，功能也逐步完善。和 Kohana 一样，Yii 的发展也是欠缺成熟的，但是 Yii 的设计思想比较超前，例如针对 jQuery 做优化等等也是其他框架不具备的。Yii 是一个很实用的框架，特别是面向中小项目，因为 Yii 所内置的类库已经足够满足一个中等规模的 Web 2.0 应用调用，而不需要再作扩展了。然而现阶段应用于大型项目，也许还需要斟酌。

Zend 由于受到 Zend 公司的支持，应该说只要 PHP 还存在，Zend 框架就会活着。同时，Oracle、IBM 等也因为各自业务的原因，支持 Zend 框架。而且对于一些大型项目来说，企业级的解决方案，也只有 Zend 这个半官方的 PHP 框架可供选择。因此，Zend 框架的前景是比较好的，也就是说，即使这个框架发展得不好，理论上它也一定还会存在并发展下去。

学习曲线

CodeIgniter 的文档齐全，结构清晰，查阅方便。对于接触的用户，官方还提供一个简单的视频教程，按照这个教程，能够很方便的开发自己的第一个 CodeIgniter 应用，并在这个过程中，熟悉 CodeIgniter 的逻辑。中等级别的用户可以查阅详细的官方文档，如下图，高级用户可以访问官方的 wiki 提供更详尽的疑难解答。CodeIgniter 是这是个架构中，学习曲线最平缓，耗时最短的。

Kohana 和 Yii 由于是社区开发的，和一般的社区开源作品一样，文档不全是个通病。你几乎很难找到 Kohana 实用的文档，连官方的文档都是版本老旧，接口错乱的。Yii 相对来说文档多一些，但是也分布得比较乱，缺乏整体条理，给上手用户带来不便。我几乎是看着 CodeIgniter 的文档，对比 CodeIgniter 和 Kohana 的差异来学习 Kohana 的。客观说，Kohana 的学习曲线比较陡峭（如果有 CodeIgniter 基础会好一点），Yii 次之。

Zend 的文档数量庞大，几乎每一个接口每一个类库都有文档支持。但是文档大而不全，缺乏代码范例，大多数类库用法几乎没有范例，只是陈述技术细节居多，所以即使有相当数量的文档，却难以上手。而且由于 Zend 框架的庞大，概念和规则比较多，所以 Zend 的学习曲线相对其他框架更加陡峭。

附录 1：PHP 开发 IDE 的选择

俗话说：“工欲善其事，必先利其器”。因此，在开始工作之前，确定好使用什么工具是很重要的，找准一个适合自己，适合项目的工具，才能事半功倍。由于项目是基于 PHP 做开发，因此我把比较流行的 PHP IDE（集成开发环境）都试用体验了一下，下面附上简短的评测笔记。

Eclipse for PHP Developers

官方网站：http://www.eclipse.org/downloads/
操作系统：Mac, Windows, Linux
软件类型：开源软件
简单备注：

IMB 推荐的开发工具，前景比较看好。
综合型开发工具，扩展便利，通过扩展支持 PHP，官方有专为 PHP 配置好的版本。
界面合理，美观，但是不如 NetBeans，使用流畅，稳定，支持对象浏览和文件浏览方式。高强度使用从未出现错误。基于 Java。
调试功能强大。
可以通过插件在 IDE 中实现各种版本管理，但是配置比较麻烦。

Komodo

官方网站：http://www.activestate.com/komodo/
操作系统：Mac, Windows, Linux
软件类型：商业软件
简单备注：

界面简单清爽（缺少对象查看器），程序运行很稳定，支持 Macros 脚本，可以内置浏览器调试，每当保存文件的时候，浏览器就自动刷新，调试小程序挺方便的。

phpDesigner

官方网站：http://www.mpsoftware.dk/
操作系统：Windows Only
软件类型：专业版收费，个人版免费
简单备注：

界面比较乱（可能因为界面和 Windows7 不兼容的缘故），没出现不稳定的情况，有对象查看器，有 xdebug 调试器，仅支持 svn 版本管理，但是程序很臃肿，不好使。

Phped

官方网站：http://www.nusphere.com/
操作系统：Windows Only
软件类型：商业软件
简单备注：

界面美观，但功能分布凌乱。稳定性表现还算满意，但是功能并不出色，而且扩展不便。
带调试功能，内置 PHP 执行引擎。

PHPEdit

官方网站：http://www.phpedit.com/
操作系统：Windows Only
软件类型：商业软件
简单备注：

此软件在实际使用中频繁出现不稳定，意外关闭的情况，因此放弃继续试用！

Zend Studio

官方网站：http://www.zend.com/en/products/studio/
操作系统：Mac, Windows, Linux
软件类型：商业软件
简单备注：

Zend 开发，前景比较看好。实际上也是基于 Eclipse 开发的，所以 Eclipse 的特性他都有，同时有一个比较完整的 php 手册。

NetBeans

官方网站：http://www.netbeans.org/
操作系统：Mac, Windows, Linux

软件类型：开源软件
简单备注：

Oracle 支持开发，前景比较看好。
综合型开发工具，扩展便利，通过扩展支持 PHP，官方有专为 PHP 配置好的版本。
界面优雅，使用流畅，稳定，支持对象浏览和文件浏览方式。高强度使用从未出现错误。基于 Java。
调试功能完善。
集成多种版本管理（hg、cvs、svn 等）！十分便利！
整合 Bug 跟踪系统，而且在 commit 的时候可以选择同时标注消灭一个 Bug。

Dreamweaver

官方网站：http://www.adobe.com/
操作系统：Mac, Windows
软件类型：商业软件
简单体验：

Dreamweaver 的界面设计合理，便于定制，但缺少对象浏览器。HTML、CSS 文件编辑功能强大，可以作为一个补充充当前台文件编辑的角色。
应该说 Dreamweaver 在页面 HTML 编辑的功能很不错，只是写后后台代码的功能不怎么样。
Dreamweaver 调试功能很简单，几乎没有多大作用。但是 Dreamweaver 有数据库连接模块，可以透视数据据库的结构。版本管理做得也不好，仅支持 Adobe 自己的封闭协议。