第 10 卷第 4 期　　安徽水利水电职业技术学院学报　　Vol.10 No.4
2010 年 12 月　JOURNAL OF ANHUI TECHNICAL COLLEGE OF WATER RESOURCES AND HYDROELECTRIC POWER　Dec.2010

# 基于 PHP 技术的 MVC 框架的实现与应用

何永太
（安徽水利水电职业技术学院，安徽 合肥　231603）

摘　要：通过对 MVC 设计模式原理的分析，利用 PHP 技术，提出一种实现 MVC 框架的方法，以期实现提高大中型 PHP 项目的开发效率、增加项目的可维护性，实验验证，其方法可行。
关键词：PHP；MVC；框架
DOI：10.3969/j.issn.1671－6221.2010.04.004
中图分类号：TP312　　文献标识码：A　　文章编号：1671-6221(2010)04-0010-03

## Realization and application of MVC framework using PHP technology

HE Yong-tai，　ZHAO Yan-ping
（Anhui Technical College of Water Resources and Hydroelectric Power，Hefei 231603，China）

**Abstract**：Through analyzing the principles of MVC design pattern，using PHP technology，a method is presented that realize MVC framework. The aim of this method is to improve development efficiency about medium-sized PHP project and to increase the maintainability of the project. The feasibility of the method is verified through examples.
**Key words**：PHP；MVC；framework

PHP(PHP：Pypertext Preprocessor 超文本预处理程序)技术是当前在中小型 Web 应用开发中广泛使用的技术。但随着应用程序规模的增大，开发者会发现 PHP 在项目控制方面显得不够，团队协同开发困难。本文结合 PHP 技术实现一个 MVC(Model－View－Controller)模式的 PHP 开发框架，以弥补 PHP 本身的一些不足。

## 1　MVC 设计模式

MVC 是一种软件设计模式，它强制性使应用程序的输入、处理和输出分开，分解成模型、视图、控制器 3 种部件，它们各自处理自己的任务，如图 1 所示。

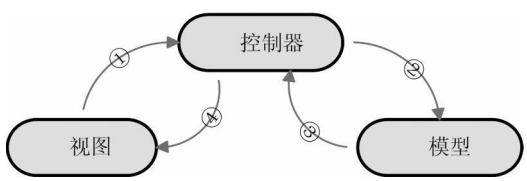

图 1　MVC 模式结构

视图(View)是用户看到并与之交互的界面。模型(Model)用来表示企业数据和业务规则，数据的处理、逻辑和功能的计算放于其中。

控制器(Controller)用来接受用户的操作并调用模型和视图完成用户的需求。

收稿日期：2010-03-16；修回日期：2010-04-06
作者简介：何永太(1977－)，男，安徽怀宁人，硕士，讲师，从事计算机软件开发技术、神经网络方面的教学与研究。

使用 MVC 模式开发 Web 应用的优点主要有：① 加快项目的开发效率；② 增强项目的可维护性；③ 利于软件工程化管理。

应用 MVC 的不足主要有：① 增加了系统的复杂性；② 在提高大型项目开发效率的同时，降低了小型项目的开发效率；③ 数据处理流程复杂，从而在一定程度上稍微降低了系统的性能。

## 2　基于 PHP 技术实现 MVC 框架

在 PHP 容器中实现 MVC 框架应达到的基本目标如下：框架易于部署；不应限制原生 PHP 特性的应用；框架具有开放性，能够在其中加入第三方的元素；应能显著提高项目的开发效率，增加项目的可维护性。基于以上目标，结合 PHP 技术本身特点，框架设计时重点从以下几点入手：

(1)单一入口的实现。单一入口是指在一个 Web 应用中，所有的请求都指向一个脚本文件。

通过单一入口方式，把所有的参数初始化、全局常量设置、安全认证、数据过滤等操作放在入口文件中；或放入独立文件中，在入口文件中进行调用。这样，在每次资源请求时都会首先执行入口程序，完成初始化等规定操作。单一入口的实现既可以通过.htaccess 文件设置规则来解决，也可以在 URL 中主动进行调用。

(2)控制器、视图和模型的交互处理。为了提供基础的 MVC 框架支持，提高代码的重用度，按照 MVC 设计模式，充分利用 PHP 的面向对象功能，把控制器和模型设计成框架所必须的核心类。

控制器类主要提供 3 类方法的支持：① loadModel( $modelName)。该方法实现装载模型，并可通过模型名称调用模型对象的方法。② loadView( $viewName)。该方法实现在浏览器窗口显示视图，并传递动态数据到视图中。③ loadLib( $libName)。该方法实现装载其它系统库或第三方库文件(也可以是用户自定义库文件)，以提供框架的扩展能力。

模型类是主要提供与数据库交互的能力，在其中实现数据库的自动链接与释放、数据库的统一操作方法，关于数据逻辑方面的处理由其子类实现。数据库的操作方法主要包括 2 大类：① query( $sql)。该方法支持原生的 SQL 语句操作，该特性保持原生 PHP 操作数据库能力；② CRUD 类操作方法：以更简易的方式提供数据库的操作能力(无需熟悉 SQL 语句)。

视图就是一个网页，或是网页的部分，如头部、底部、侧边栏等等，视图从不直接调用，必须被一个控制器来调用。视图基于现有的 PHP 程序文件，无需封装成类。

各器件间的相互协调关系如图 2 所示，说明如下：① index.php 作为前端控制器，初始化运行 MVC 框架所需要的基本资源。② Route 检查 HTTP 请求，确定选择什么控制器以及调用该控制器哪个方法。③ 安全(Security)。控制器(Controller)装载之前，HTTP 请求和任何用户提交的数据将被过滤，防止如 SQL 注入攻击等类似的安全隐患。④ 控制器(Controller)装载模型、核心库以及其它第三方的库或资源。⑤ Model 进行企业业务逻辑处理，期间可以访问数据库(DB)进行数据的读写操作。⑥ Model 处理后的数据返回到控制器，再由控制器传入视图(View)。最终视图(View)渲染发送到 Web 浏览器中的内容。

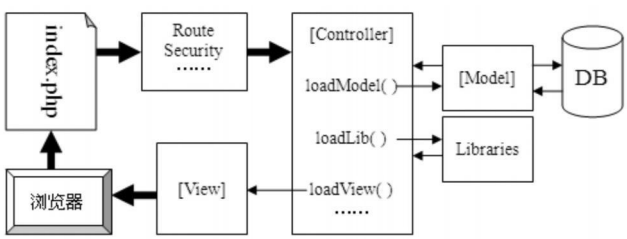


图 2　应用程序数据流程

(3)其它系统库文件的支持。这些库文件中定义了常见的操作函数功能，以提高代码的复用率，减轻开发者的负担。各种系统库文件在需要时由控制器动态载入。动态载入的目的是为了最大限度地减小对 PHP 程序执行性能的负面影响，因为不需要时不载入。

## 3 MVC 框架的应用与效果分析

应用本文的 MVC 框架进行应用开发时应遵循一定的约定，其中很重要的一条是用户自定义的控制器类和模型类必须继承自框架核心类 Controller 类和 Model 类。以下是应用本框架实现的一个用户登录功能程序，程序文件如下：

(1)loginForm.php。视图文件，用于显示简单的登录表单，内容如下：

```
<form action="login/doLogin" method="post">用户名:<input type="text" name="username"/><br/>密码:<input type="password" name="password"/><input type="submit" value="Login"/></form>
```

(2)loginResult.php。视图文件，用于显示登录的结果，内容如下：

```
<?php echo $msg;//变量 msg 的值由控制器传入?>
```

(3)login.php。控制器类文件，内容如下：

```
<?php class Login extends Controller{//index 方法为缺省执行的方法，本例用来装载登录表单
  function index(){ $this->loadView('loginForm');}
  function doLogin(){//doLogin 方法在登录表单提交时被调用
    $this->loadModel('checkLogin');//装载模型 checkLogin
    $msg= $this->checkLogin->check();//check 方法用于登录验证，并返回验证结果
    $data=array('msg'=> $msg);
    $this->loadView('loginResult', $data);//验证结果传入被转载的视图 loginResult 中
}}?>
```

(4)checkLogin.php。模型类文件，实现登录的验证算法，代码如下：

```
<?php class CheckLogin extends Model{
  function check(){if(isset($_POST['username']) && isset($_POST['password']) &&
    $_POST['username']=='hyt' && $_POST['password']=='123'){return '合法登录';}
     else{return '用户名或密码错误';}
}}?>
```

通过以上案例可看出，本框架很好的把用于前端显示的视图与数据处理的后端分离开来，这非常有利于项目的分工，前台开发人员可以专注于页面的设计，程序员可专注于企业业务逻辑处理的设计；同时也非常有利于后期的维护，如登录验证算法改变了，则只需修改 checkLogin.php 文件，其它文件不受任何影响。当项目规模增大时，这种好处会越发得到体现。

## 4 结束语

实践表明，在基于 PHP 技术进行的项目开发中，应用本文所提出的 MVC 框架，确实起到了显著提高开发效率、利于团队协作、便于项目维护、灵活适应需求变更的效果。同时也并没有增加对开发者的技术要求。

[参 考 文 献]

[1] Kevin McArthur 著. PHP 高级程序设计. 模式、框架与测试[M]. 汪泳 译. 北京：人民邮电出版社，2009.

[2] 赵增敏. PHP 动态网页开发[M]. 北京：电子工业出版社，2009.

[3] 李 宁. Java Web 开发技术大全[M]. 北京：清华大学出版社，2009.

[4] Ted Husted, et al. Struts in Action, Building Web Application with the Leading Java Framework[M]. Manning Publications Co., 2003.

(责任编辑 胡 进)