博客系统第1天

[学习目标 2](#_Toc522896909)

[项目开发流程 2](#_Toc522896910)

[1、需求分析 2](#_Toc522896911)

[2、概要设计 2](#_Toc522896912)

[3、详细设计 2](#_Toc522896913)

[4、编码工作 2](#_Toc522896914)

[5、测试工作 3](#_Toc522896915)

[6、交付和用户验收 3](#_Toc522896916)

[7、后期维护 3](#_Toc522896917)

[博客系统概述 3](#_Toc522896918)

[1、博客系统介绍 3](#_Toc522896919)

[2、导入博客系统数据库 3](#_Toc522896920)

[3、运行博客项目 4](#_Toc522896921)

[4、博客项目的后台功能 5](#_Toc522896922)

[5、博客项目的前台功能 5](#_Toc522896923)

[博客项目的MVC目录结构 6](#_Toc522896924)

[博客项目MVC框架搭建 7](#_Toc522896925)

[1、前台配置文件：./Home/Conf/Config.php 7](#_Toc522896926)

[2、前台入口文件：index.php 7](#_Toc522896927)

[3、框架核心类文件：./Frame/Frame.class.php 8](#_Toc522896928)

[4、测试前端MVC框架是否运行正常 8](#_Toc522896929)

[学生信息管理(前台) 9](#_Toc522896930)

[1、首页控制器：./Home/Controller/IndexController.class.php 9](#_Toc522896931)

[2、首页模型类：./Home/Model/IndexModel.class.php 9](#_Toc522896932)

[3、数据库工具类：./Frame/Libs/Db.class.php 10](#_Toc522896933)

[学生信息管理(后台) 10](#_Toc522896934)

[1、后台配置文件：./Admin/Conf/Config.php 10](#_Toc522896935)

[2、后台入口文件：admin.php 11](#_Toc522896936)

[3、首页控制器：./Admin/Controller/IndexController.class.php 11](#_Toc522896937)

[4、首页模型类：./Admin/Model/IndexModel.class.php 12](#_Toc522896938)

[封装PDOWrapper类 12](#_Toc522896939)

[1、为什么要封装PDOWrapper类 12](#_Toc522896940)

[2、创建PDOWrapper类：./Frame/Vendor/PDOWrapper.class.php 13](#_Toc522896941)

[PDOWrapper类与MVC框架整合 13](#_Toc522896942)

[1、创建基础模型类：./Frame/Libs/BaseModel.class.php 13](#_Toc522896943)

[2、其它模型类继承基础模型类 14](#_Toc522896944)

[工厂模型类方法的实现 14](#_Toc522896945)

[1、基础模型类添加getInstance()静态方法 14](#_Toc522896946)

[2、首页控制器类index()方法修改 15](#_Toc522896947)

[封装自己的Smarty类 15](#_Toc522896948)

[1、创建自己的Smarty类：./Frame/Vendor/Smarty.class.php 15](#_Toc522896949)

[2、创建基础控制器类：./Frame/Libs/BaseController.class.php 16](#_Toc522896950)

[3、首页控制器index()方法修改 16](#_Toc522896951)

[4、首页视图文件：index.html 17](#_Toc522896952)

# 学习目标

* 项目开发流程
* 博客项目介绍
* 博客项目目录结构规划
* 博客项目MVC框架搭建
* 封装PDOWrapper类
* 封装Smarty类

# 项目开发流程

## 1、需求分析

系统分析员向用*户初步了解需*求，用相关的工具软件列出系统的大功能模块，每个大功能模块有哪些小功能模块。并且还要列出相关的界面和界面功能。向用户再次确认需求。

## 2、概要设计

首先，开发者需要对项目进行概要设计，即系统设计。包括项目的基本处理流程、组织结构、模块划分、功能分配、接口设计、运行设计、数据结构设计和出错处理设计等，为项目的详细设计提供基础。

## 3、详细设计

在概要设计的基础上，进行项目的详细设计。包括：主要算法、数据结构、类的层次结构及调用关系等，以及每个函数的每个参数的定义都精精细细的提供出来。详细设计应当足够详细。一份详细设计说明书，一个项目就已经完成了一半工作了。换句话：在项目完成一半工作时，还没有开始写一行代码。

## 4、编码工作

根据《项目详细设计报告》中对数据结构、算法分析和模块实现等方面的设计要求，开始具体的编写程序工作，分别实现各模块的功能，从而实现对目标系统的功能、性能、接口、界面等方面的要求。在规范化的研发流程中，编码工作在整个项目流程里最多不会超过1/2，通常在1/3的时间。所以设计过程完成的好，编码效率就会极大提高。

## 5、测试工作

测试编写好的系统，确认每个功能。软件测试有很多种：按照测试执行方，可以分为内部测试和外部测试；按照测试范围，可以分为模块测试和整体联调。总之，测试同样是项目研发中一个相当重要的步骤，对于一个大型软件，3个月到1年的外部测试都是正常的。完成测试后，整体项目才算告一段落。

## 6、交付和用户验收

在软件测试证明软件达到要求后，应向用户提交开发的目标安装程序、数据库的数据字典、《用户安装手册》、《用户使用指南》、需求报告、设计报告、测试报告等双方合同约定的产物。

## 7、后期维护

根据用户需求的变化或环境的变化，对应用程序进行全部或部分的修改。

# 博客系统概述

## 1、博客系统介绍

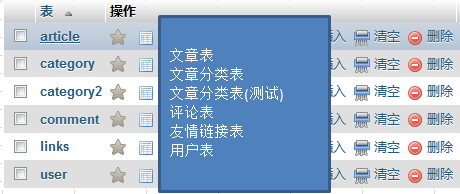
* 博客，仅音译，英文名为Blogger,为Web Log(网络日志)的混成词。
* 博客就是以网络作为载体，简易迅速便捷地发布自己的心得，及时有效轻松地与他人进行交流，再集丰富多彩的个性化展示于一体的综合性平台。
* Blog是继Email、BBS、ICQ之后出现的第四种网络交流方式，至今已十分受大家的欢迎，是网络时代的个人“读者文摘”，是以超级链接为武器的网络日记，是代表着新的生活方式和新的工作方式，更代表着新的学习方式。
* 博客上的文章通常根据张贴时间，以倒序方式由新到旧排列。
* 博客可以包含文字、图像、音乐、视频、链接等网络要素内容。
* 著名的博客有：新浪博客(blog.sina.com.cn)、网易博客(blog.163.com)等。

## 2、导入博客系统数据库

**使用phpMyAdmin导入blog.sql数据库备份文件**



**博客系统的数据表介绍**

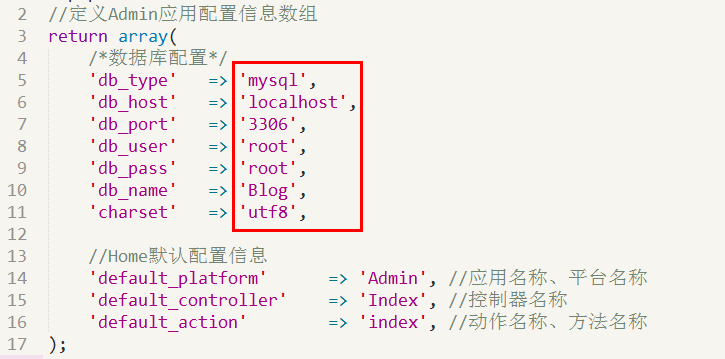


## 3、运行博客项目

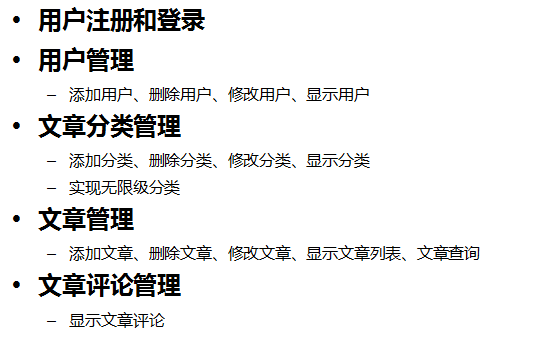
修改博客项目前端配置：./Home/Conf/Config.php



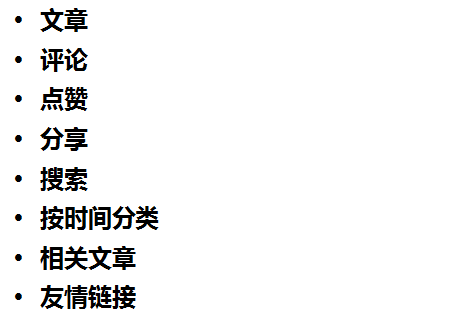
修改博客项目后端配置：./Admin/Conf/Config.php



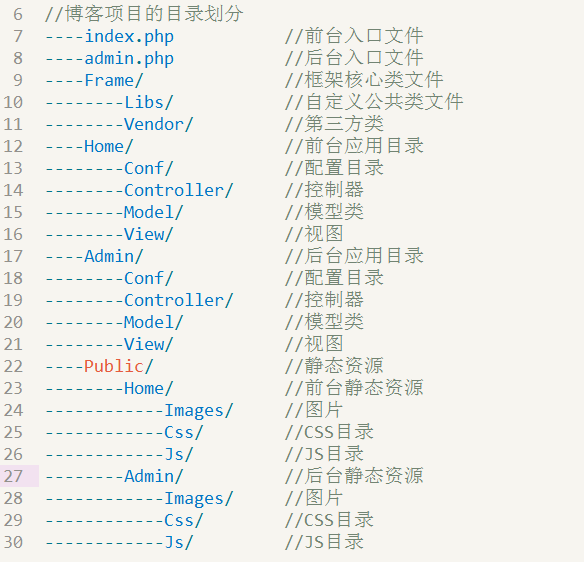
## 4、博客项目的后台功能



## 5、博客项目的前台功能

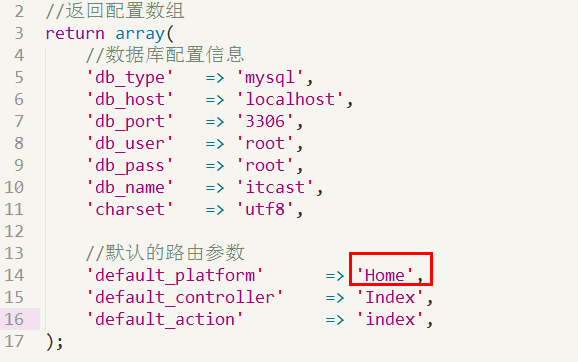


# 博客项目的MVC目录结构

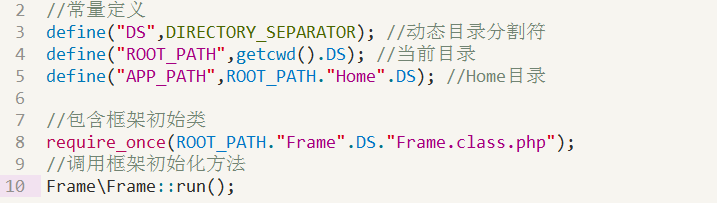


# 博客项目MVC框架搭建

## 1、前台配置文件：./Home/Conf/Config.php



## 2、前台入口文件：index.php



## 3、框架核心类文件：./Frame/Frame.class.php



## 4、测试前端MVC框架是否运行正常



# 学生信息管理(前台)

## 1、首页控制器：./Home/Controller/IndexController.class.php



## 2、首页模型类：./Home/Model/IndexModel.class.php

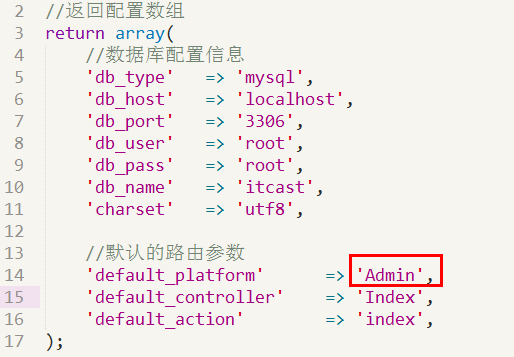


## 3、数据库工具类：./Frame/Libs/Db.class.php

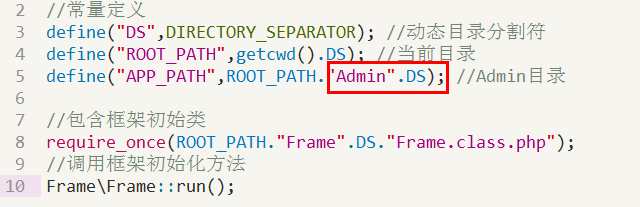


# 学生信息管理(后台)

## 1、后台配置文件：./Admin/Conf/Config.php



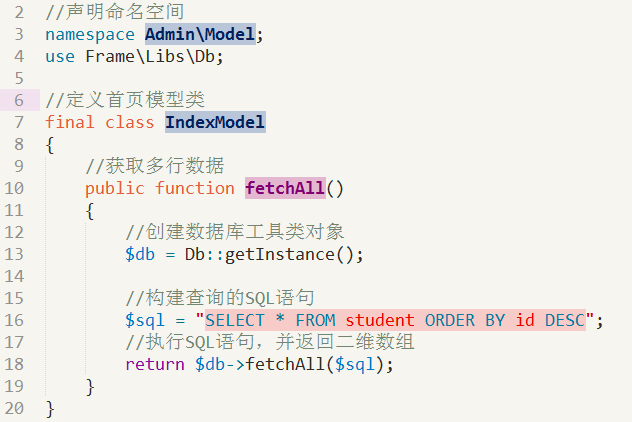
## 2、后台入口文件：admin.php



## 3、首页控制器：./Admin/Controller/IndexController.class.php



## 4、首页模型类：./Admin/Model/IndexModel.class.php



# 封装PDOWrapper类

## 1、为什么要封装PDOWrapper类

* 博客项目中，所有类都定义了命名空间，系统的PDO没有定义命名空间；
* 对PDO中方法再次封装，可以简化操作；
* 对PDO的异常处理进行封装，方便对PDO错误处理统一管理。

## 2、创建PDOWrapper类：./Frame/Vendor/PDOWrapper.class.php



# PDOWrapper类与MVC框架整合

## 1、创建基础模型类：./Frame/Libs/BaseModel.class.php

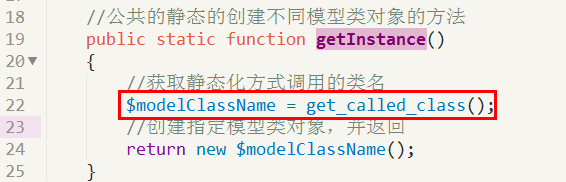


## 2、其它模型类继承基础模型类



# 工厂模型类方法的实现

## 1、基础模型类添加getInstance()静态方法



## 2、首页控制器类index()方法修改



# 封装自己的Smarty类

## 1、创建自己的Smarty类：./Frame/Vendor/Smarty.class.php

**提示：把原始的Smarty文件夹，复制到 ./Frame/Vendor/ 目录下。**



## 2、创建基础控制器类：./Frame/Libs/BaseController.class.php



## 3、首页控制器index()方法修改



## 4、首页视图文件：index.html

