MVC框架第2天

[学习目标 2](#_Toc522810524)

[昨日回顾 2](#_Toc522810525)

[1、MVC框架思想 2](#_Toc522810526)

[2、MVC流程图 3](#_Toc522810527)

[MVC第8个版本：基础控制器类实现 3](#_Toc522810528)

[1、需求分析 3](#_Toc522810529)

[2、创建基础控制器类：BaseController.class.php 4](#_Toc522810530)

[3、其它控制器继承基础控制器 4](#_Toc522810531)

[MVC第9个版本：学生信息修改 4](#_Toc522810532)

[1、需求分析 4](#_Toc522810533)

[2、用户操作起点：StudentIndexView.html 5](#_Toc522810534)

[3、学生控制器添加edit()方法 5](#_Toc522810535)

[4、学生模型类添加fetchOne()方法 5](#_Toc522810536)

[5、修改学生信息的视图文件：StudentEditView.html 6](#_Toc522810537)

[6、学生控制器添加update()方法 7](#_Toc522810538)

[7、学生模型类添加update()方法 7](#_Toc522810539)

[MVC第10个版本：文件简单分目录 8](#_Toc522810540)

[1、需求分析 8](#_Toc522810541)

[2、MVC目录简单划分 8](#_Toc522810542)

[3、修改index.php 9](#_Toc522810543)

[4、修改控制器文件中包含视图文件的路径 9](#_Toc522810544)

[MVC第11个版本：前端控制器实现 9](#_Toc522810545)

[1、需求分析 9](#_Toc522810546)

[2、修改index.php文件 10](#_Toc522810547)

[3、修改视图文件导航栏地址和JS跳转地址 11](#_Toc522810548)

[4、修改控制器方法中的跳转到地址 11](#_Toc522810549)

[MVC第12个版本：平台概念的引入 11](#_Toc522810550)

[1、需求分析 11](#_Toc522810551)

[2、MVC最终的目录结构 12](#_Toc522810552)

[3、修改index.php文件 13](#_Toc522810553)

[4、修改视图文件导航栏地址和JS跳转地址 14](#_Toc522810554)

[5、修改控制器方法中跳转地址 15](#_Toc522810555)

[MVC第13个版本：类的自动加载 16](#_Toc522810556)

[1、需求分析 16](#_Toc522810557)

[2、修改index.php文件 16](#_Toc522810558)

[MVC第14个版本：常用目录常量配置 16](#_Toc522810559)

[1、需求分析 16](#_Toc522810560)

[2、修改index.php文件 17](#_Toc522810561)

[MVC第15个版本：初始类实现 17](#_Toc522810562)

[1、修改index.php文件 17](#_Toc522810563)

[2、配置文件：./App/Conf/Config.php 18](#_Toc522810564)

[3、创建初始框架类文件：./Frame/Frame.class.php 18](#_Toc522810565)

[4、修改基础模型类文件：./Frame/BaseModel.class.php 19](#_Toc522810566)

[5、修改数据库工具类文件：./Frame/Db.class.php 19](#_Toc522810567)

# 学习目标

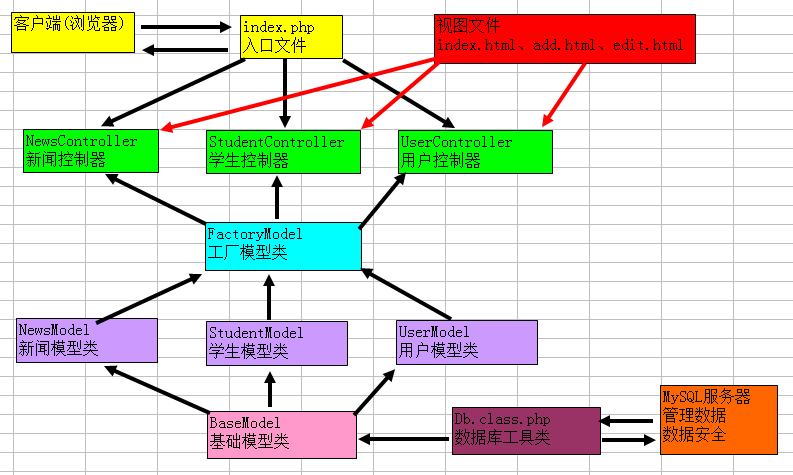
* MVC第8个版本：基础控制器类实现
* MVC第9个版本：修改学生信息
* MVC第10个版本：文件简单分目录
* MVC第11个版本：前端控制器实现
* MVC第12个版本：平台概念引入
* MVC第13个版本：类的自动加载
* MVC第14个版本：常用目录常量配置
* MVC第15个版本：框架初始类实现

# 昨日回顾

## 1、MVC框架思想

* MVC就是编程思想，是一种软件设计的典范，没有任何新知识点；
* MVC是一种新的代码的组织方式；
* MVC是由Model、View、Controller三个组件构成；
* Controller控制器：与客户打交道、程序业务逻辑、调用Model处理数据、调用View展示数据。
* Model数据模型：主要负责与数据库打交道，包括：增加、删除、修改、查询等。
* View展示数据：对数据进行格式化处理(包装)，主要涉及技术：HTML、CSS、JS、Flash等。
* 一个项目由若干个功能模块构成：学生管理、新闻管理、用户管理；
* 每个功能模块对应一个控制器类：StudentControll、NewsController、UserController
* 每个控制器对应一个模型类：StudentModel、NewsModel、UserModel
* 每个模型类对应一张数据表的操作：student、news、user
* 每个控制器对应多个视图：StudentIndexView.html、StudentAddView.html、StudentEditView.html

## 2、MVC流程图



# MVC第8个版本：基础控制器类实现

## 1、需求分析

将多个子控制器的公共代码，提取出来，放到基础控制器类中，其它控制器再继承基础控制器。

基础控制器类的好处：减少重复代码的编写、方便后期维护、类的层次更加清晰。



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



## 3、其它控制器继承基础控制器



# MVC第9个版本：学生信息修改

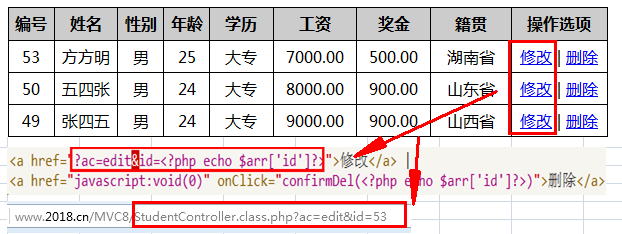
## 1、需求分析

在MVC中，修改学生信息，大致分两步：

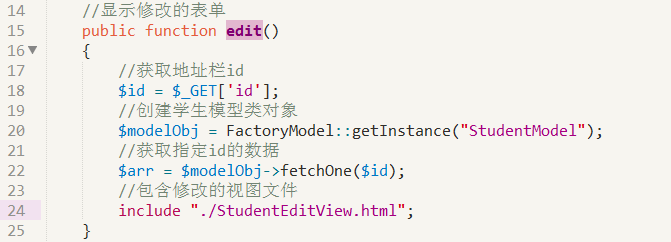
第1步：显示修改的表单，读取指定学生信息，并填写表单中；

第2步：提交修改的表单，并完成数据的更新工作。

## 2、用户操作起点：StudentIndexView.html



## 3、学生控制器添加edit()方法



## 4、学生模型类添加fetchOne()方法



## 5、修改学生信息的视图文件：StudentEditView.html





## 6、学生控制器添加update()方法



## 7、学生模型类添加update()方法



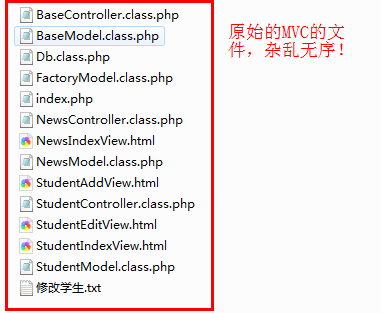
# MVC第10个版本：文件简单分目录

## 1、需求分析

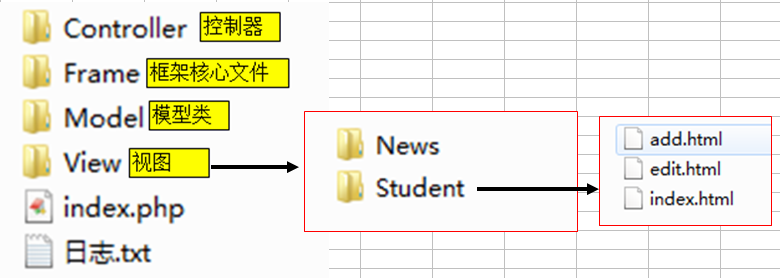
首先，将多个模块的控制器文件、模型类文件、视图文件放在根目录下，显得杂乱无序，不方便管理。

**按照MVC思想，对所有文件进行简单目录划分**：

* Model目录：主要存放各个模块的模型类文件；
* Controller目录：主要存放各个模块的控制器类文件；
* View目录：主要存放各个模块的视图文件；
* Frame目录：主要存放各个模块公共的类文件。



## 2、MVC目录简单划分



## 3、修改index.php

将所有类的包含语句，全部挪到index.php页面。



## 4、修改控制器文件中包含视图文件的路径



# MVC第11个版本：前端控制器实现

## 1、需求分析

* 前端控制器(请求分发器)，就是指index.php！
* 它的作用就是：决定使用哪个控制器，以及使用哪个方法？
* 前端控制器有了之后，在每次请求中，都应该包含两个信息：控制器名和动作名。
* 习惯上，我们不使用像action或controller长长的名称；
* 控制器名，可使用c来代替；动作名，可用a来代替；
* 如果没有指定控制器或动作，请用默认控制器或动作来代替。
* 请求格式：index.php?c=控制器名&a=动作名&其它参数
* 举例说明：index.php?c=Student&a=edit&id=100

## 2、修改index.php文件





## 3、修改视图文件导航栏地址和JS跳转地址



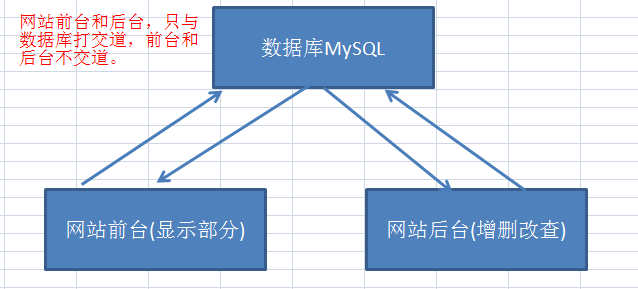
## 4、修改控制器方法中的跳转到地址



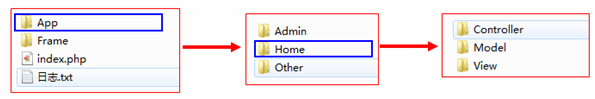
# MVC第12个版本：平台概念的引入

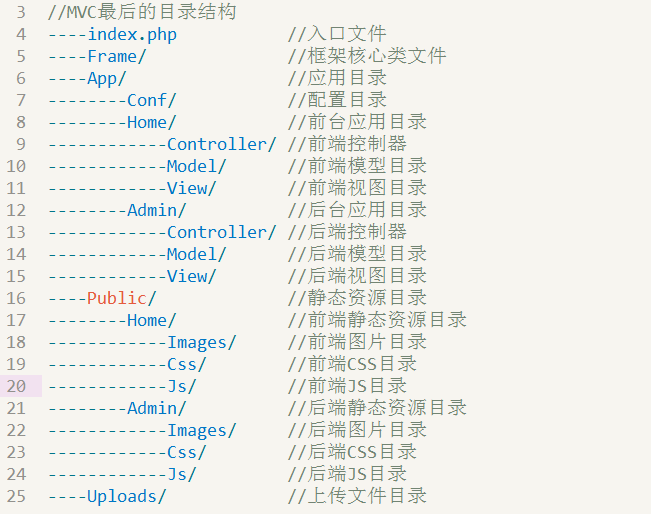
## 1、需求分析

* 平台，就是“前台”“后台”“合作方平台”等这样一些概念的总称。
* 在一个MVC框架结构中，往往都可能有多个相对独立的应用站点，最典型的就是一个网站的前台界面(对外的)，和后台管理界面（对内的）。
* 一个平台，就是一个相对独立的应用（站点，项目）。
* 有了“平台”概念后，MVC的目录结构，就要进一步扩展了。
* 引入“平台”概念后，客户的每次请求，必须带3个参数：平台参数、控制器参数、用户动作。
* Platform称为p参数，controller称为c参数，action简称a参数。
* 请求地址：index.php?p=平台&c=控制器&a=动作&其它参数
* 举例说明：**index.php?p=Home&c=Student&a=index**



## 2、MVC最终的目录结构





## 3、修改index.php文件





## 4、修改视图文件导航栏地址和JS跳转地址



## 5、修改控制器方法中跳转地址





# MVC第13个版本：类的自动加载

## 1、需求分析

如果客户每次请求，都去包含所有类文件，会造成服务器内存浪费，程序的加载速度会更加缓慢。

我们可以使用类的自动加载，来实现类文件的按需加载，不需要的不用加载。

## 2、修改index.php文件



# MVC第14个版本：常用目录常量配置

## 1、需求分析

常量在任何地方都可以使用，而变量只能在其作用域下使用；

我们把MVC中常用的目录定义成常量，可以简化目录路径的编写。

## 2、修改index.php文件



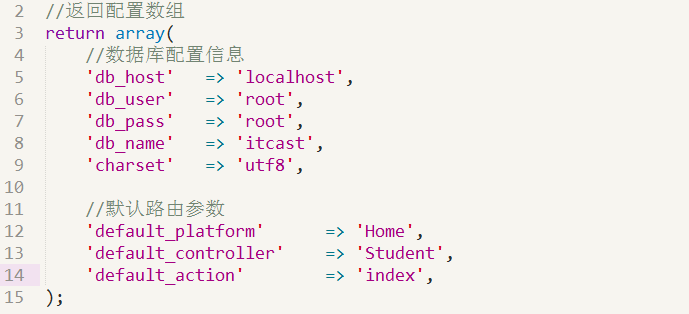


# MVC第15个版本：初始类实现

## 1、修改index.php文件



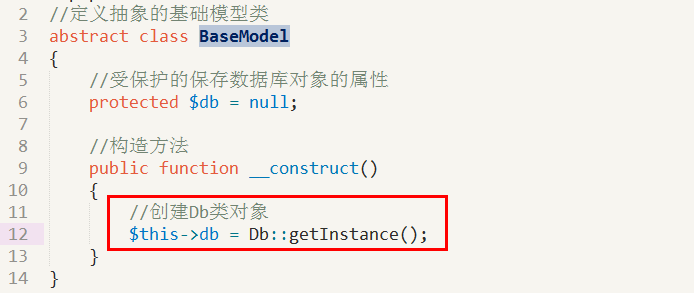
## 2、配置文件：./App/Conf/Config.php



## 3、创建初始框架类文件：./Frame/Frame.class.php



## 4、修改基础模型类文件：./Frame/BaseModel.class.php



## 5、修改数据库工具类文件：./Frame/Db.class.php

