1. 项目架构规划

二、项目时间规划

|  |  |  |
| --- | --- | --- |
| 时间 | 任务 | 备注 |
| 第一天 | 搭建mvc框架、数据库设计 | 整体架构 |
| 第二天 | 商品品牌模块、商品分类模块 | 后台功能 |
| 第三天 | 权限控制、项目安全、商品类型模块 |
| 第四天 | 商品属性模块、商品模块 |
| 第五天 | 前台首页、商品页、列表页、上线 | 前台功能 |
| 第六天 | 完善、测试、答辩 | 完善 |

1. 编程思想
2. OOP思想 - 面向对象编程

封装 ： 隐藏内部实现的细节，给外部提供一个接口

继承 ： 代码重用

多态 ： 多种形态

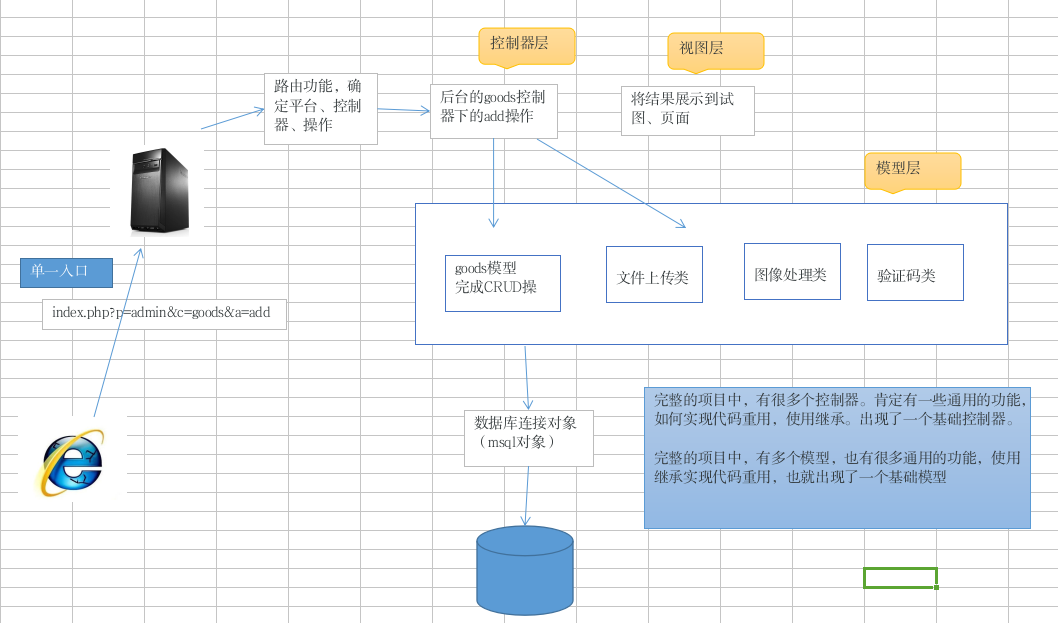
OOP好处： 好维护，可重用，可扩展，灵活度高

1. MVC思想 - 业务逻辑和显示相分离

M ： Model - 模型，完成具体的业务逻辑，包括数据库的CRUD

V ： View - 视图，显示用户的交互页面

C ： Controller - 控制器，负责业务调度



1. 项目搭建

设计要点：

->良好的项目结构

->统一的命名规范

->单一入口

1. 项目结构说明

项目目录：

---appication 应用程序目录

---models 模型目录

---views 视图目录

---admin 后台视图目录

---home 前台视图目录

---controllers 控制器目录

---admin 后台控制器目录

---home 前台控制器目录

---config 配置文件目录

---framework 架构目录

---core 框架核心类

---database 数据库驱动类目录，存放数据驱动类，如mysql类，pdo类

---helpers 辅助类函数目录，自定义的函数类

---libraries 工具类目录，如image类，upload类

---public 前台静态资源目录

---css

---images

---js

---uploads

---index.php 单一入口文件

1. 搭建过程
2. 编码规范说明

1、一定要有注释

2、要有统一的命名规范

文件名（类名.class.php，如Mysql.class.php，不是类就不加class）

类名、方法名、属性名

/\*

类名使用大驼峰式，如GoodsController，对应的文件名是GoodsController.class.php；

方法名使用小驼峰式，addAction；

属性名也使用小驼峰式，有时候会对private的属性前面加一个下划线。

具体编写的时候，写控制器和模型的时候都需要指明controller和model。

\*/

函数名

/\*

自定义的普通函数，不是类中的方法建议遵循php本身的写法，如var\_dump,is\_array下划线式，还有imagecreatetruecolor全小写方式，选一种。

\*/

1. 严格区分大小写
2. 注意缩进，代码对齐
3. 定义核心启动类

启动类应该实现什么？

已index.php?p=admib&c=goods&a=add请求为例，须实现：初始化工作（路径加载等），路由功能，自动加载功能

如何自动加载？

**√** 直接在index.php中编写\_\_autoload()方法;

**√** 在类中定义一个方法，然后将其注册为自动加载方法;（推荐）

1. 加载视图

加载静态页面，只需要使用include的方式即可

目标-》显示后台页面

所有相对路径的请求，都是相对index.php而言；

1. 定义基础控制器（核心控制器） -- 其他控制器都要继承它

定义所有控制器都需要的方法；

问题：基础控制器写在什么地方？

基础控制器与具体的业务无关，所有的项目都可以使用，所有应该放在framework中

基础控制器的功能： 须手动载入，直接在init方法中载入

**√**提示信息的跳转

**√**载入其他模型（工具类和辅助函数）

跳转页面： 跳转到某个页面之后，弹出提示信息，n秒跳回来 （message.html）

**√**Php中使用header函数

**√**Js中使用location.href=url

**√**Html中使用meta标签实现跳转 （meta可以控制时间，所有使用meta）

1. 载入数据库模型
2. 载入工具类和辅助类

在基础控制器中定义方法，完成工具类和辅助函数的加载

1. 数据库设计

要点：

**√**表与表之间的联系

**√**每个字段的类型

图片的保存：将图片文件本身保存到磁盘的某个目录下，然后将其路径保存到数据库的字段中；

表与表的关系： 一对一 一对多 多对多

品牌表和商品表之间的关系： 一对多

一个品牌对应n个商品

一个商品对应一个品牌

在mysql中没有布尔类型，通常使用tinyint来表示；

商品的规格参数如何保存？（难点）

由两部分组成：

**√**基础信息：所有的商品都具备的信息

**√**扩展信息：不同类型的商品，他的属性有不一样的

对于基础信息，可以直接保存到商品表中；

扩展信息如果用一个大字段一次性保存，比如用text，但是不宜管理，不利搜索；

商品 和 扩展信息 是多对多的关系（用中间表保存关系）

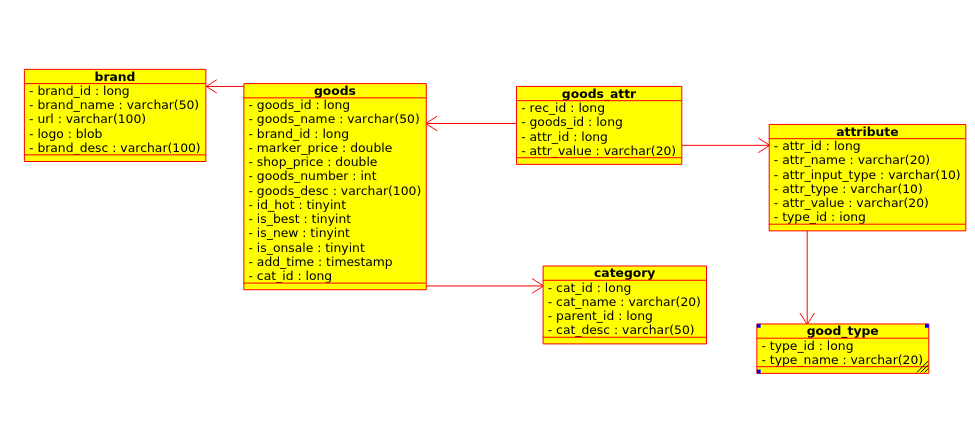
关联表的主键方式：

联合主键，使用两个字段构成主键

再加一个字段表示主键

面临新的问题： 那么多的扩展属性，如何管理？

根据商品的类型对扩展属性进行管理，增加一张“商品类型”表，表示商品扩展属性的类别。



1. 后台功能
2. 前台功能
3. 上线