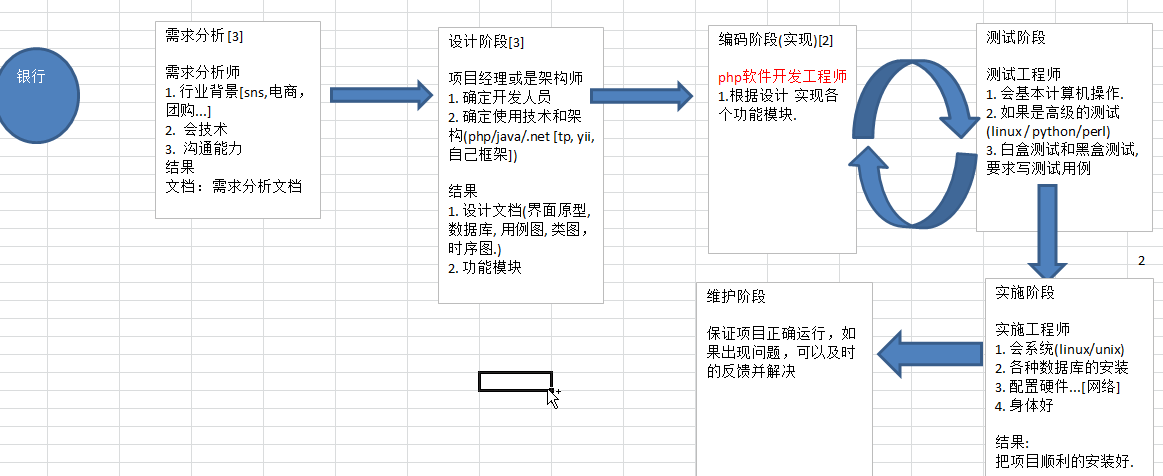
# 泰牛博客项目

# 项目的开发的流程

示意图说明: []



# 需求分析

按照标准的博客的需求分析来开发.

# 界面设计

有设计阶段提供， 前端工程师给我们.

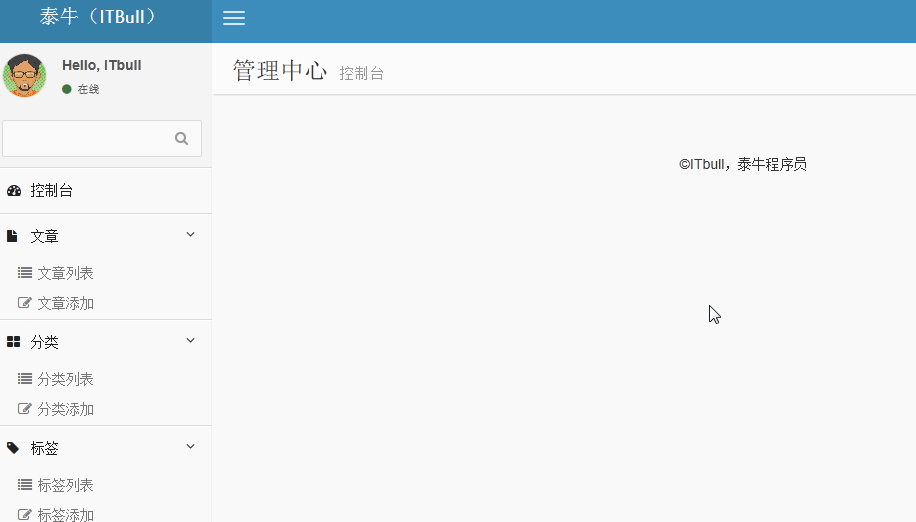
首页面:



博客的详细页



后台的管理页







# 数据库的设计

设计数据库的依据:

1. 需求分析文档
2. 原型图
3. 要适当的和其他同事(需求分析人员，项目经理)

## 文章表

create table tn\_article(

id int unsigned primary key auto\_increment comment '文章id',

title varchar(64) not null default '' comment '文章标题',

content text not null comment '文章内容',

user\_id int unsigned not null default 1 comment '用户id',

create\_ad int unsigned not null default 0 comment '发布时间',

summary varchar(256) not null default '' comment '文章简介',

category\_id int unsigned not null default 1 comment '分类id',

cover varchar(256) not null default '' comment '封面图片',

state enum('publish', 'save') not null comment '状态',

is\_delete tinyint unsigned not null default 0 comment '是否删除',

tags varchar(256) not null default '' comment '标签'

)engine = myisam charset=utf8;

//测试数据

insert into tn\_article values(null, '泰牛博客1标题', '泰牛博客1内容', 1, unix\_timestamp(), '泰牛博客1简介', 1, ' Koala.jpg', 'publish', 0, '');

insert into tn\_article values(null, '泰牛博客2标题', '泰牛博客2内容', 1, unix\_timestamp(), '泰牛博客2简介', 1, 'Penguins.jpg', 'publish', 0, '');

//测试数据

insert into tn\_article values(null, '泰牛博客3标题', '泰牛博客3内容', 1, unix\_timestamp(), '泰牛博客2简介', 2, 'Desert.jpg', 'publish', 0, '');

insert into tn\_article values(null, '泰牛博客4标题', '泰牛博客4内容', 1, unix\_timestamp(), '泰牛博客4简介', 3, 'Lighthouse.jpg', 'publish', 0, '');

## 分类表:

//等会...

## 用户表:

//等会..

## 标签表:

//等会..

## 评论表:

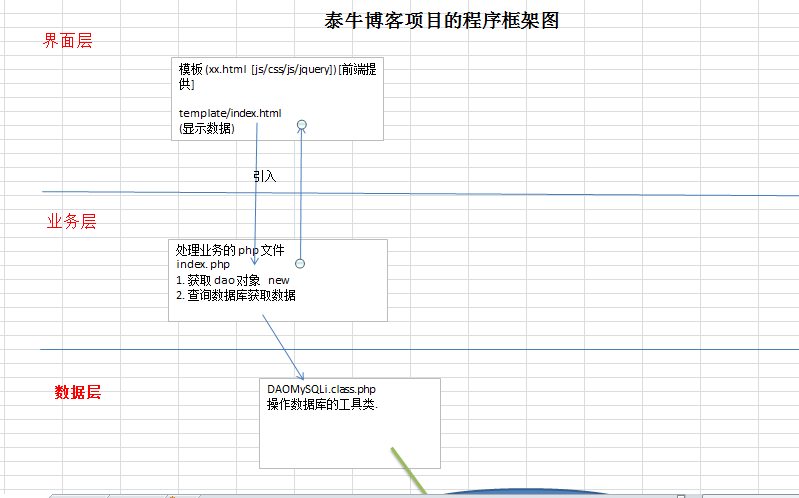
//等会..

# 规划程序框架图(三层结构[界面、业务、数据层])

//这里我们画出博客的程序框架图[说明有多少个文件，以及文件之间的调用关系]，

//确定一个设计方案，使用三层机构.

* 第一阶段图

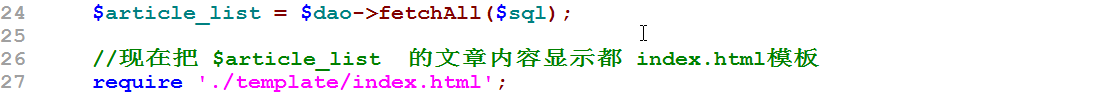


# 完成功能

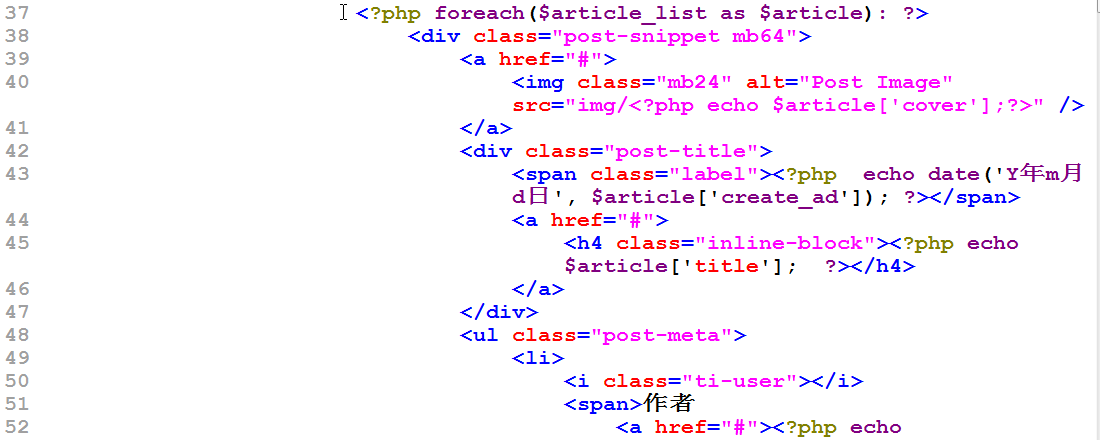
## 博客的首页面正常显示

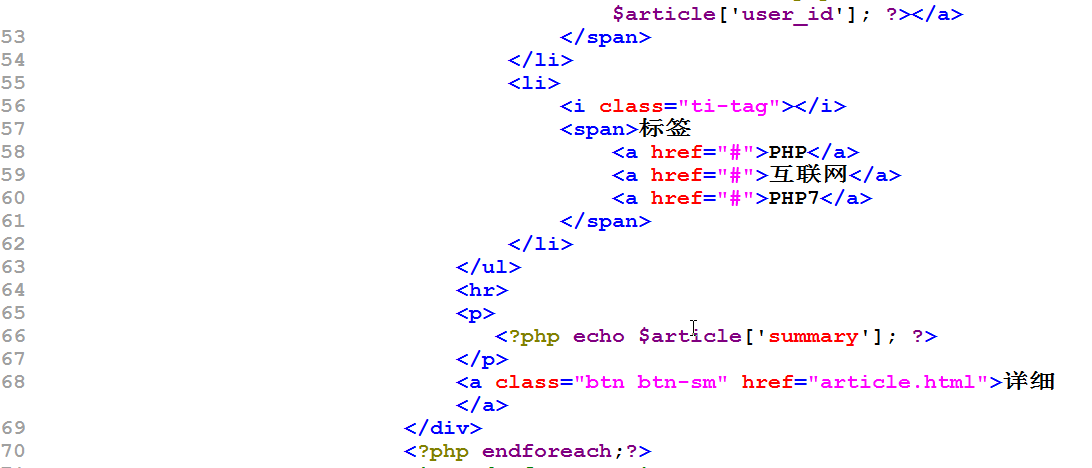
index.php



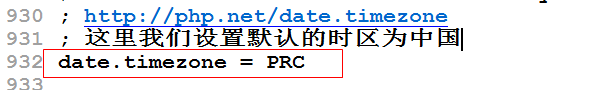


index.html





注意事项: 为了正确的显示时间，我们在php.ini 文件设置默认的时区



DAOMySQLi.class.php

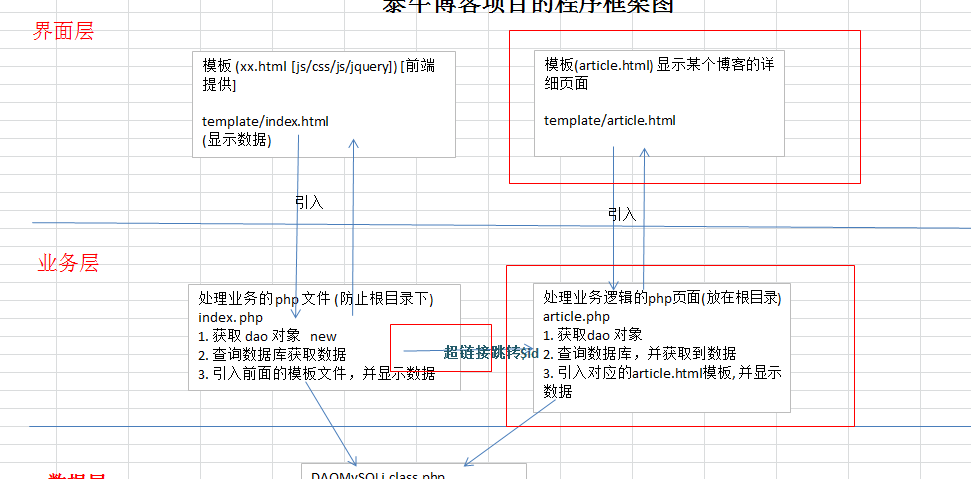
这个文件和上次课一样的.

## 当用户点击首页面的详细超链接时，显示对应的博客文章的具体信息.

即，点击详细超链接后，可以进入到对应的博客文章详细页

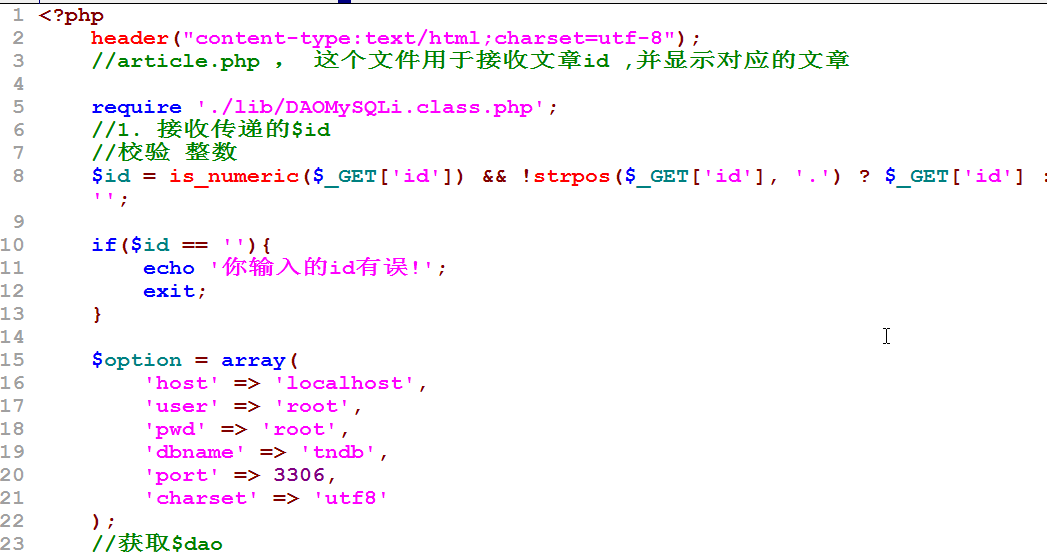


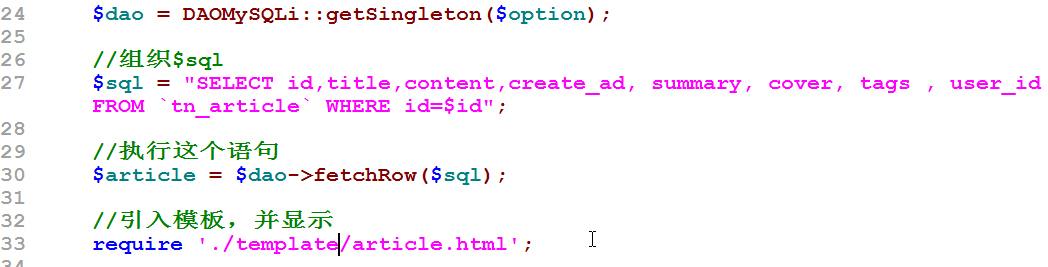
思路:



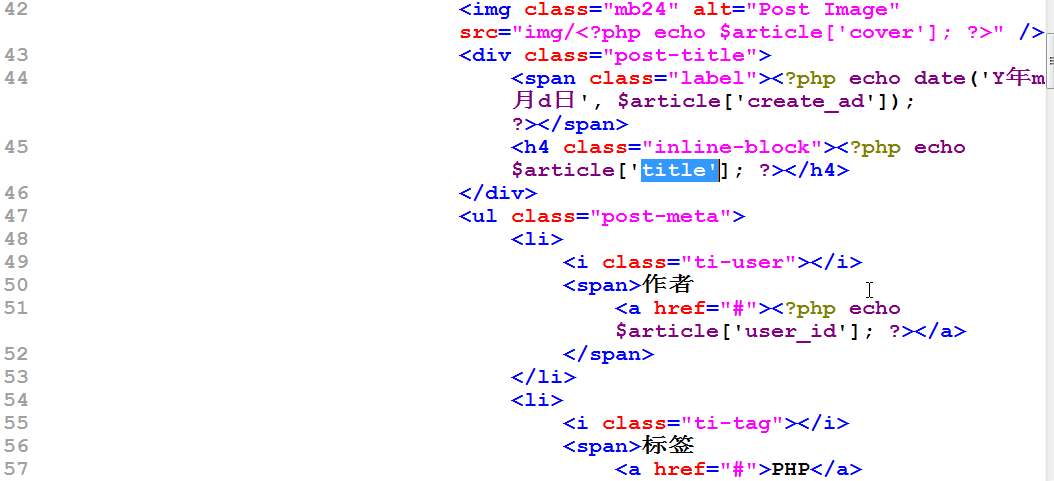
走代码:

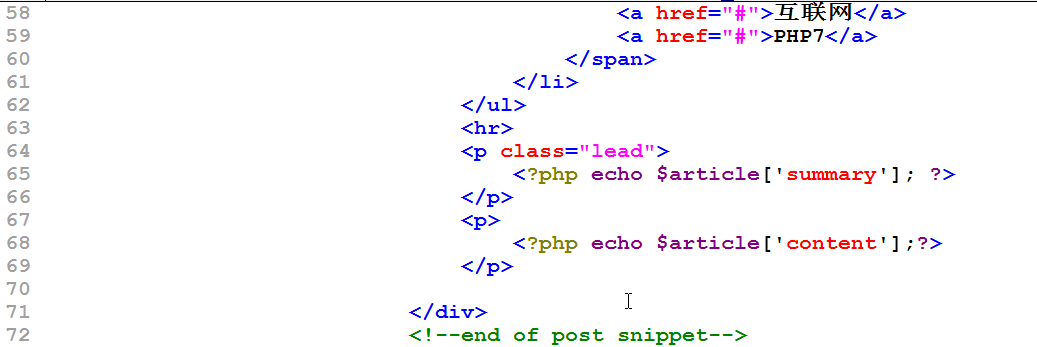
article.php



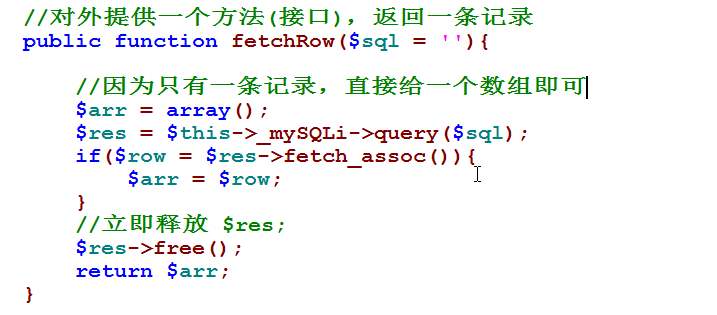


article.html





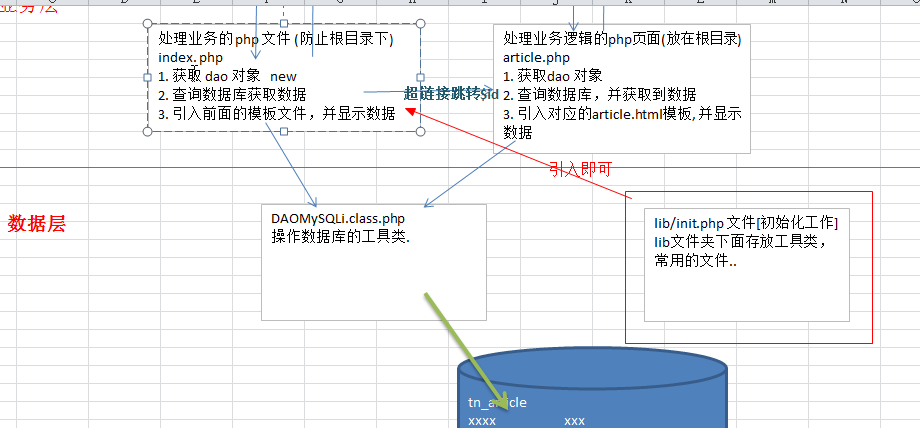
对DAOMySQLi.class.php



## 对上面的代码进行一个优化处理

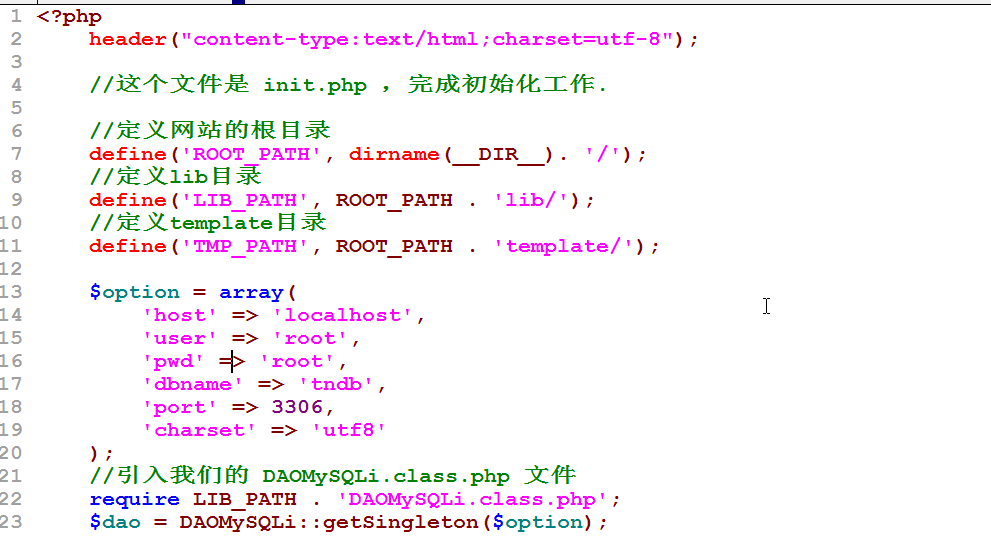
分析：因为我们的业务层的 .php 文件，几乎都要使用 $dao 对象，因此我们可以把初始化的工作，统一的写到一个 init .php 文件中, 可以统一管理，便于维护.

思路:

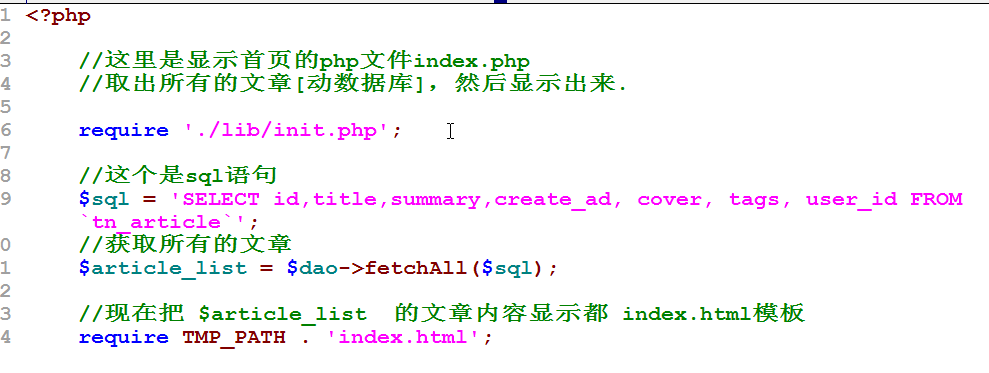


修改如下:

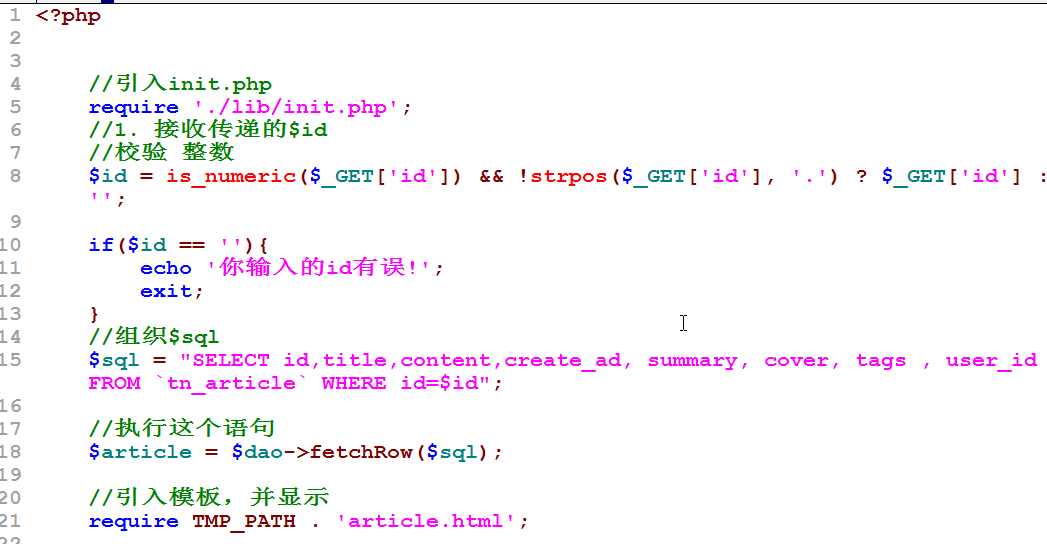
1. 增加了一个 lib/init.php



1. 对index.php 做了修改



1. 对article.php 做了修改



## 在首页面模板可以显示分类，点击分类可以显示对应类别的博客

1. 分析:

创建一个分类表, 该表可以存放所有分类.

create table `tn\_category`(

id smallint unsigned primary key auto\_increment comment '分类id',

title varchar(64) not null default '' comment '分类名称',

order\_number int unsigned not null default 100) engine=myisam charset=utf8;

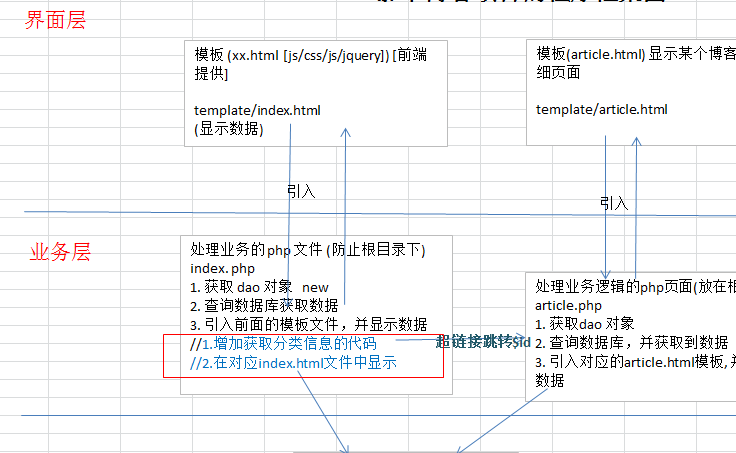
--测试数据

insert into `tn\_category` values(null, 'php技术类', 101);

insert into `tn\_category` values(null, '职场规划类', 102);

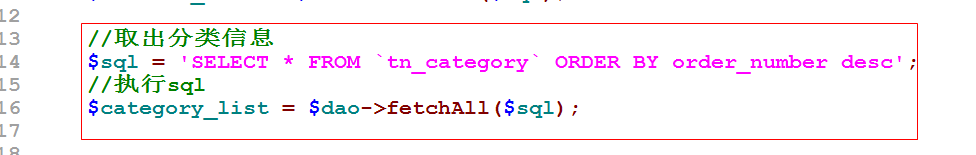
insert into `tn\_category` values(null, '数据算法类', 103);

1. 思路(图):

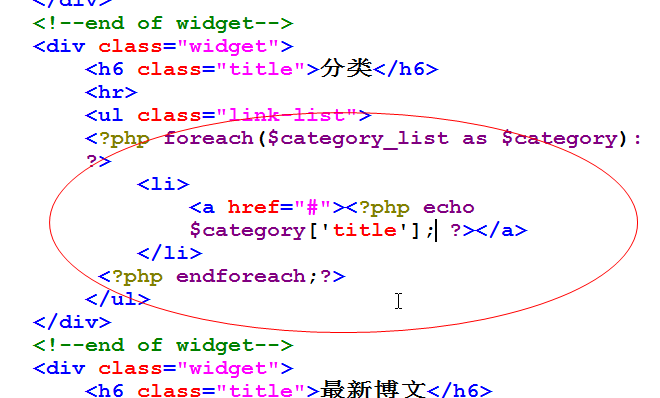


1. 走代码

* 添加一张表 tn\_category;
* 修改index.php 文件

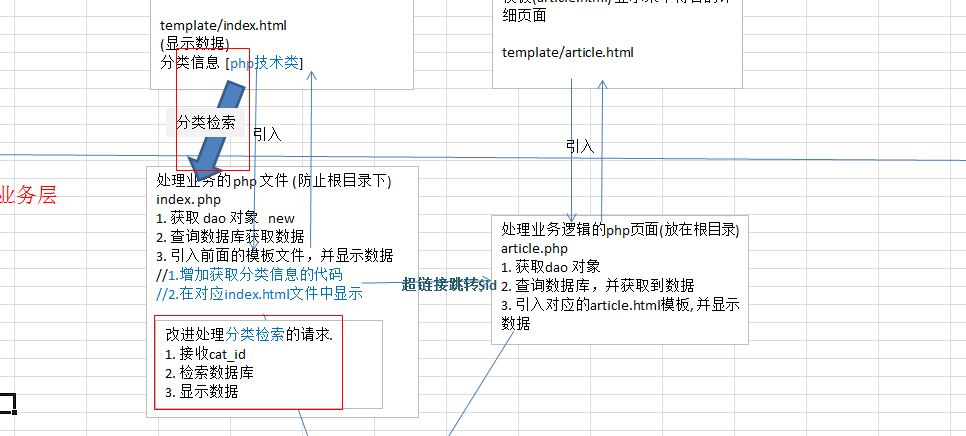


* 修改index.html模板



## 点击分类可以显示对应分类的博客文章

1. 分析

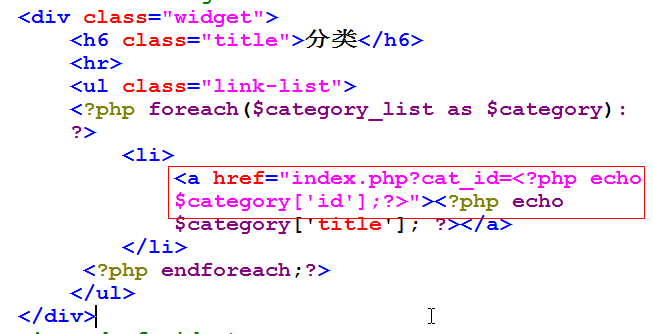


编程思路:

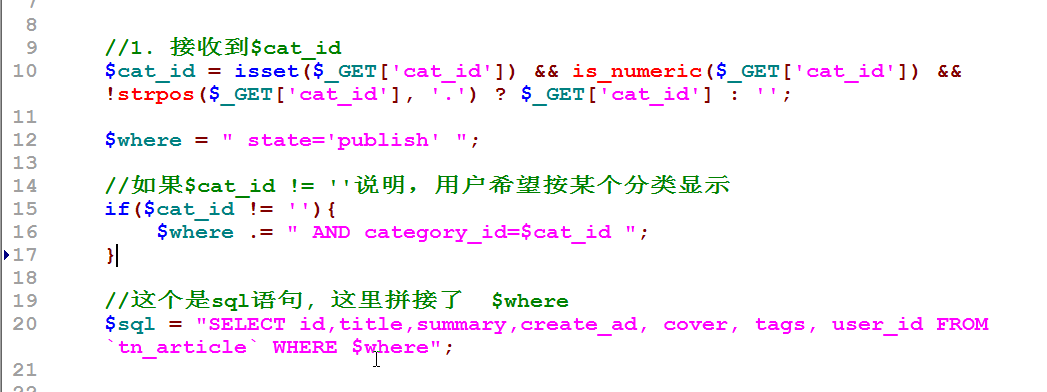
一个模板的超链接 点击 -----》 xx.php ---> 获取到$action -->判断$action 是什么--->对应处理(使用$dao 获取数据，然后显示数据.)[示意图]

1. 代码

* 修改了 index.html 模板，创建链接



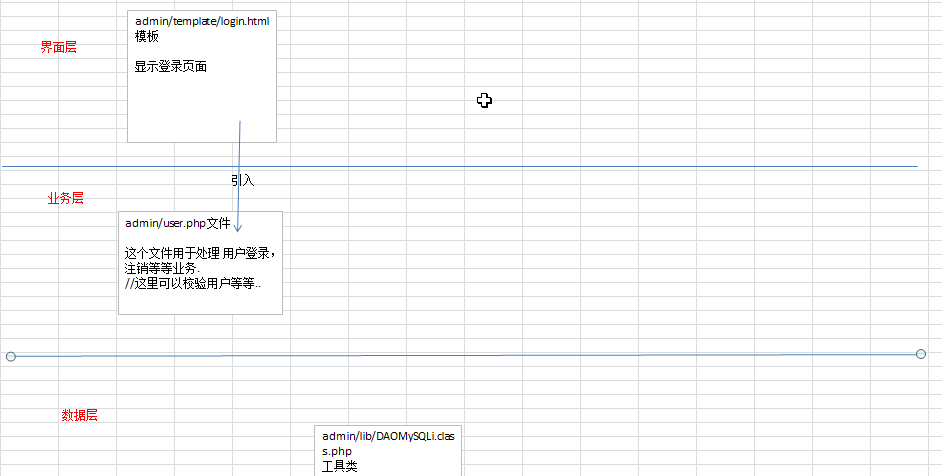
* 修改了index.php 页面



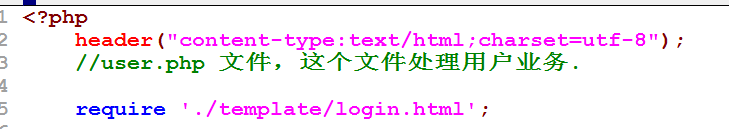
## 完成功能说明：做出管理员登录界面

1. 分析

找到前端给我们提供的模板，并引入到项目中.



1. 代码
2. 把后台拷贝到 tnblog/admin 下 这个文件夹下是我们后台相关的模板和php文件
3. 编写一个user.php 文件

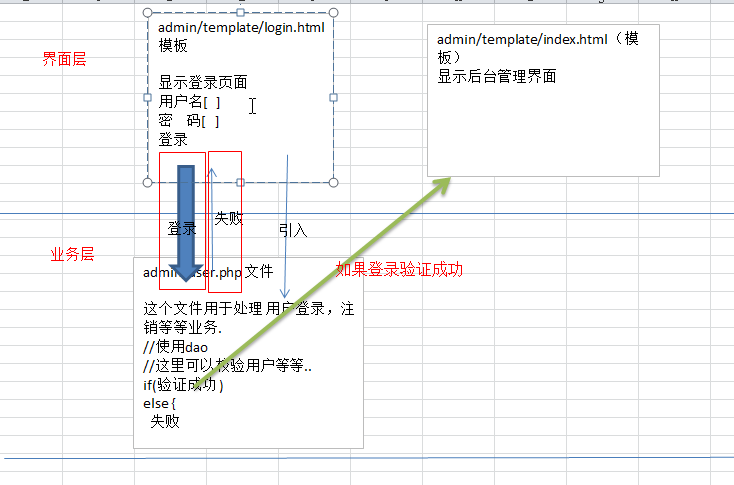


1. 这时访问 localhost/tnblog/admin/user.php 就看到登录页面



## 用户后台可以正确登录，登录成功显示后台页面，可以注销退出

1. 分析



1. 实现代码

* 创建 tn\_user表

create table `tn\_user` (

id int unsigned primary key auto\_increment comment '用户id',

username varchar(64) not null default '' comment '用户名',

password char(32) not null default '' comment '用户密码',

nickname varchar(64) not null default '' comment '用户昵称',

logo varchar(255) not null default 'default.jpg' comment '用户头像',

create\_at int unsigned not null default 0 comment '用户创建时间',

status enum('online', 'offline', 'busy', 'hidden') not null comment '用户的状态'

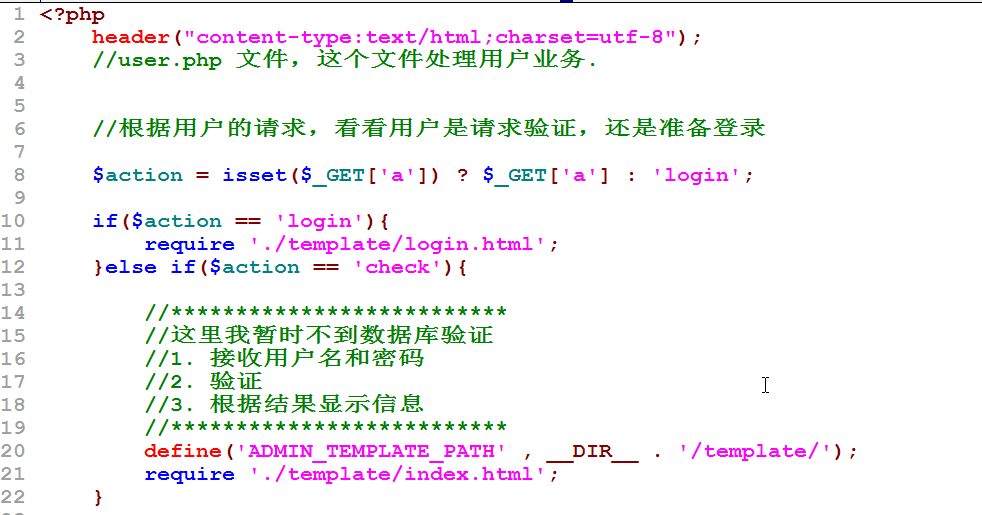
)engine=myisam charset = utf8;

--测试数据

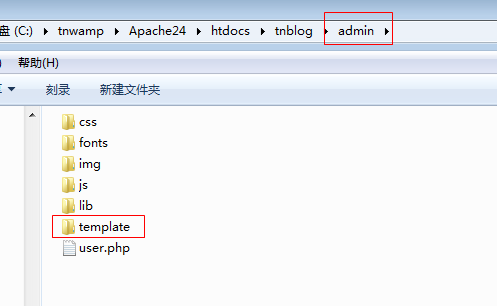
insert into `tn\_user` values(null, 'tainiu', md5('123abc'),'泰牛小圆子', 'avatar.png', unix\_timestamp(), 'online');

* 先把后台的管理页面可以正确引入[暂时不验证]

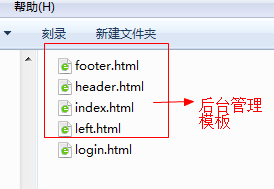
1. user.php 修改了代码



1. 我们引入了后台的管理模板

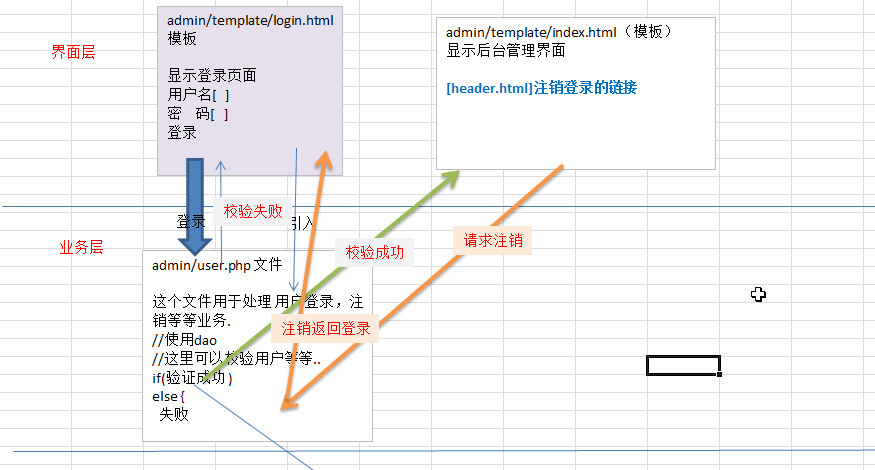


在template 目录下增加了四个模板文件



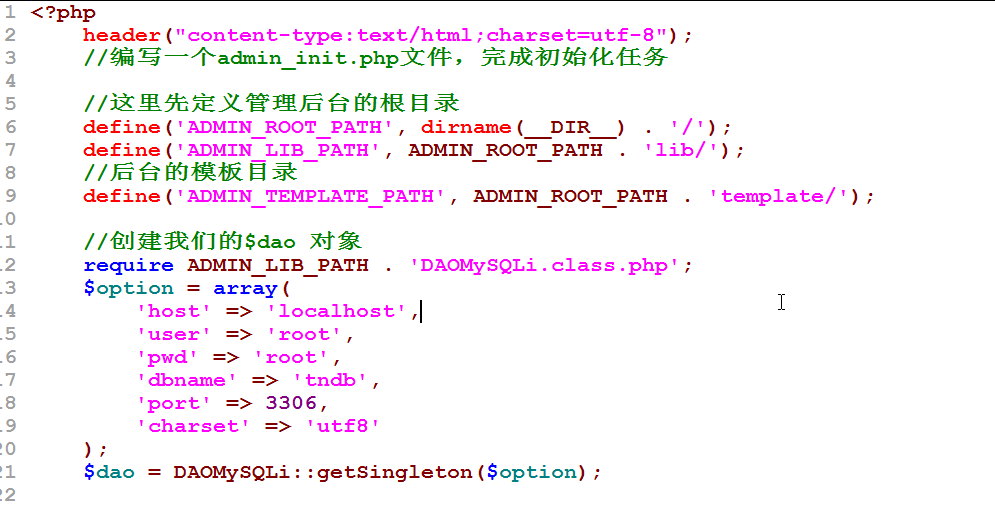
## 用户后台可以正确登录（到数据库验证），登录成功显示后台页面，可以注销退出

分析: 当我们接收到用户输入的用户名和密码后，到数据验证，根据验证的结果来决定跳转到哪个对应的页面, 因为需要使用到 $dao 对象，因此我们把 admin\_init.php 文件写引入到我们的user.php 文件.



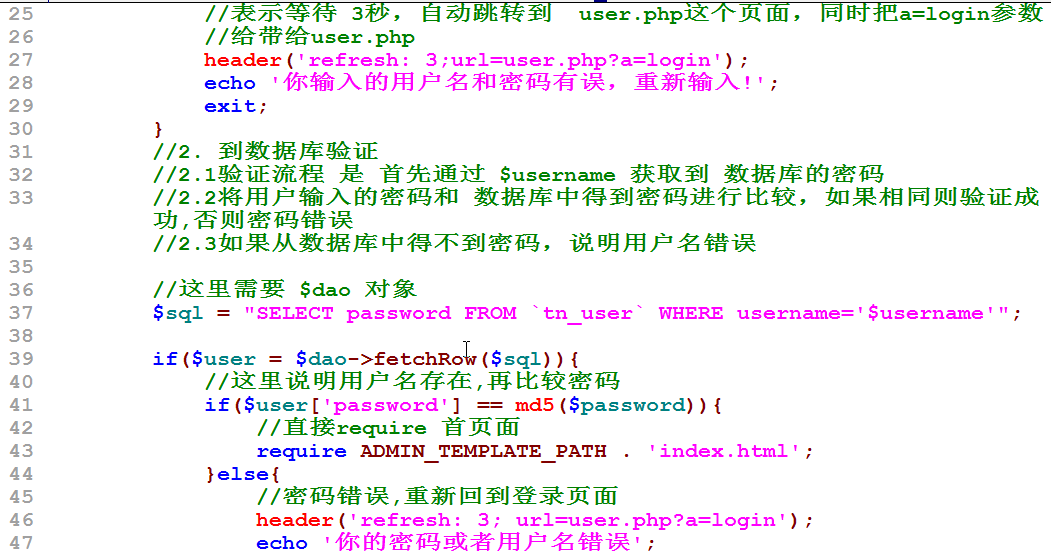
* 走代码

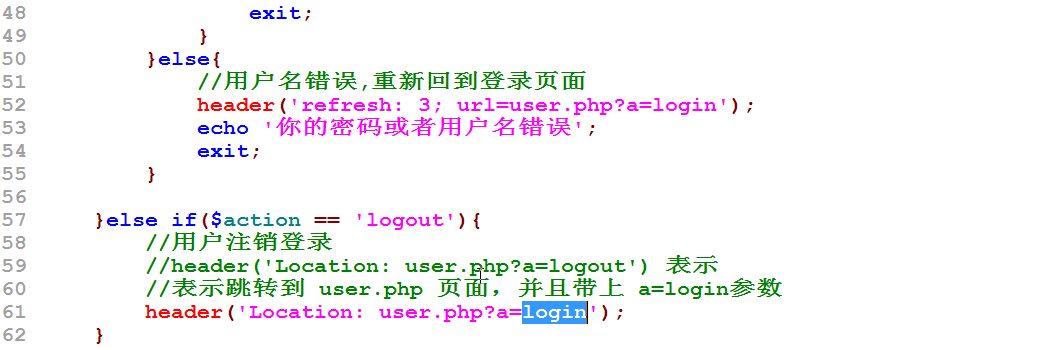
1. 增加了一个admin/admin\_init.php 文件



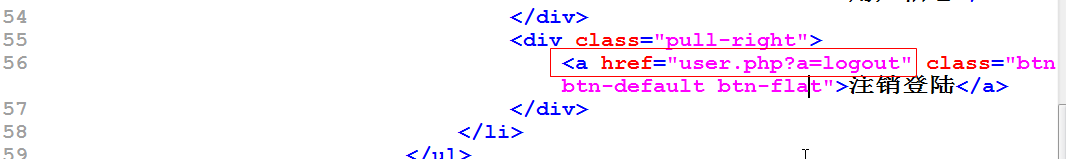
1. 在user.php 文件中增加了引入admin\_init.php 和 校验 和注销的业务代码





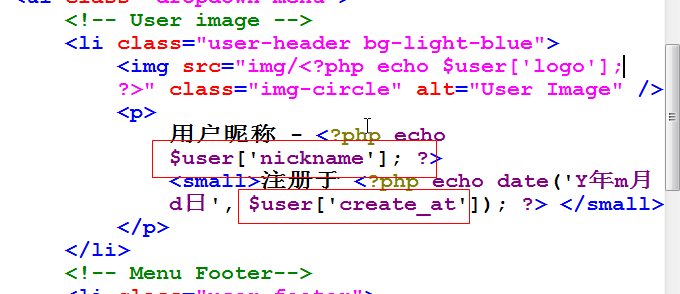


1. 在index.html 引入的 header.html文件中，增加了注销的请求链接



## 用户登录后台后，可以显示用户的各个信息

1. 分析: 这里我们直接使用了user.php 页面查询到$user对象,然后在 index.html 中的各个模板页显示。
2. 代码如下:



然后找到需要显示的位置，改成对应的数据.

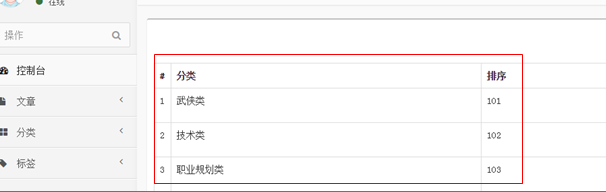
## 用户登录后台，可以点击分类列表，显示出所有的分类信息

1. 分析:

点击后台管理页面的左侧的

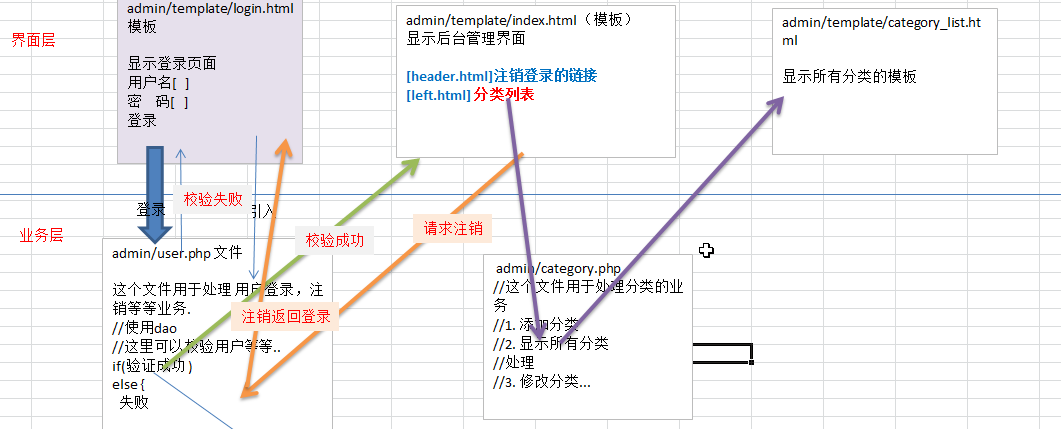


出现:



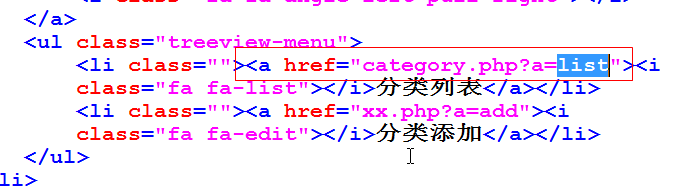
进入新的模板.

重新设计一下我们的程序框架图:



1. 走代码

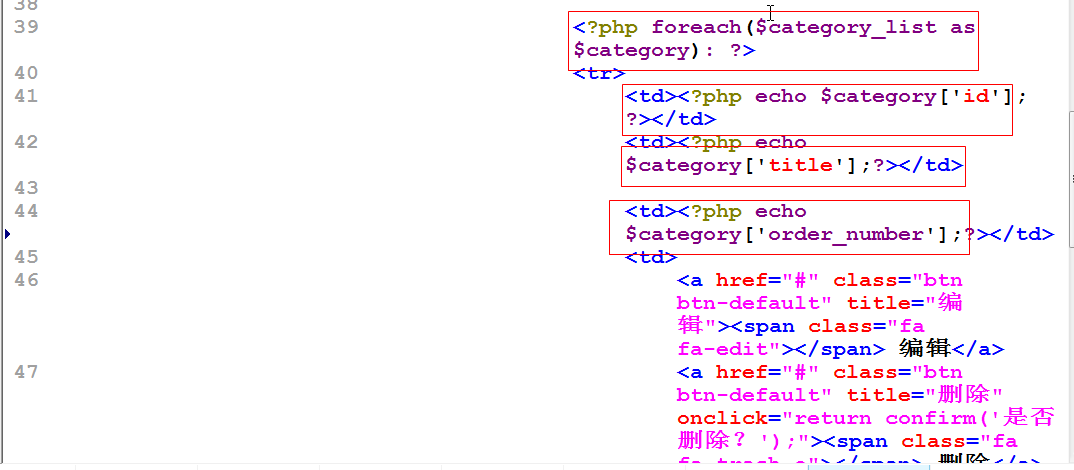
修改了left.html 这个模板

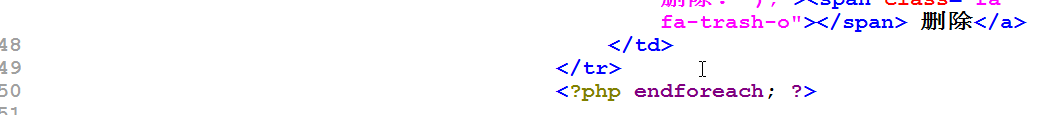


增加了一个新的文件 admin/category.php



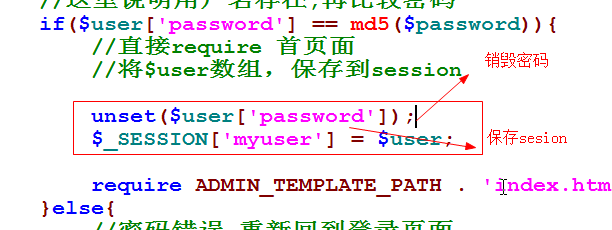
增加了一个新的模板 admin/template/category\_list.html





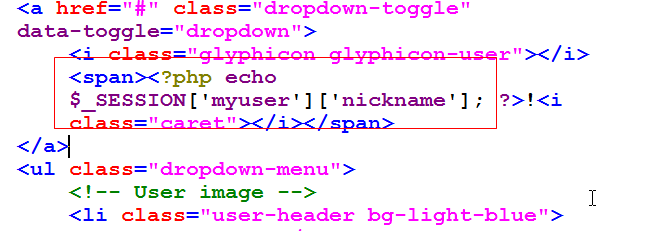
这里我们使用session 来显示登录用户的信息(入门级的介绍)

修改user.php文件.

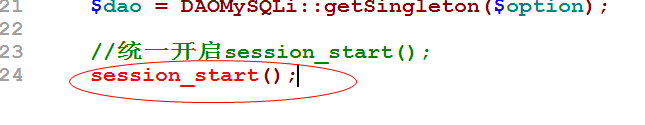


在需要读取session数据时

header.html 和 left.html

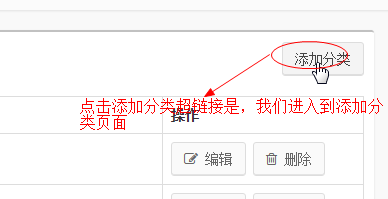


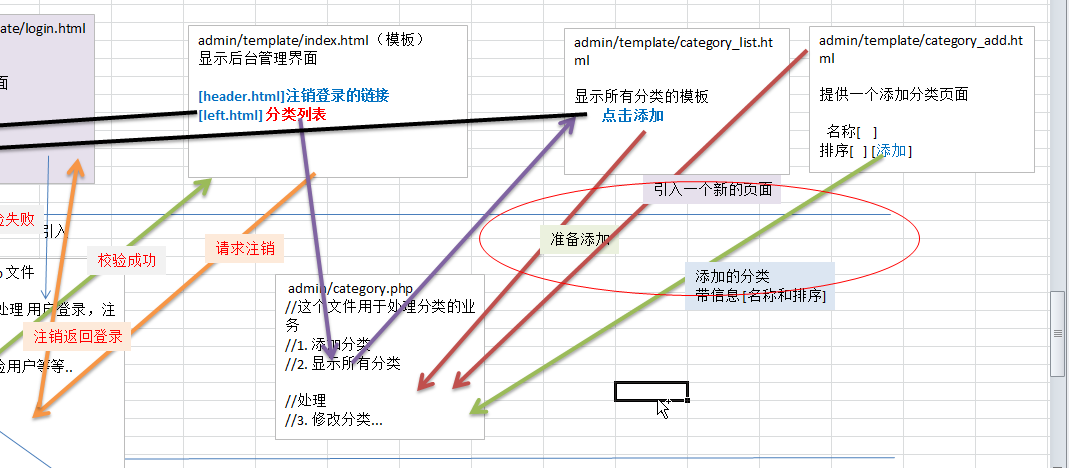
在admin\_init.php 文件统一开启了session



## 在后台可以增加新的文章分类

1.分析

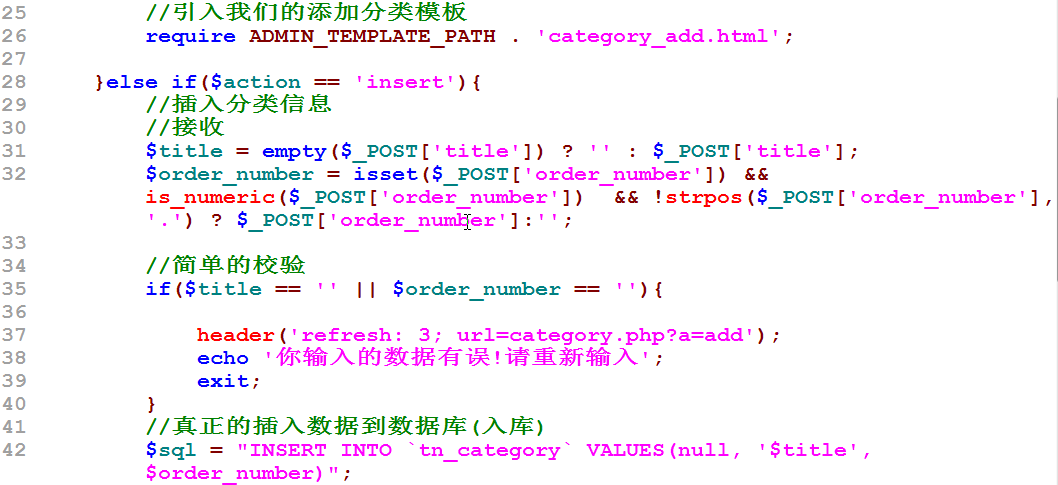


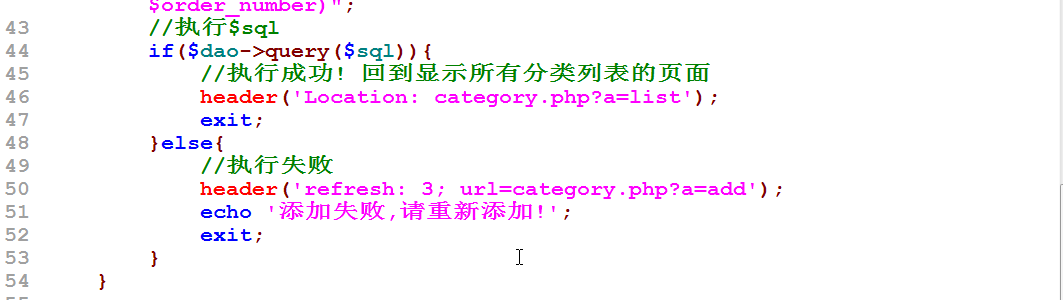


代码 ：

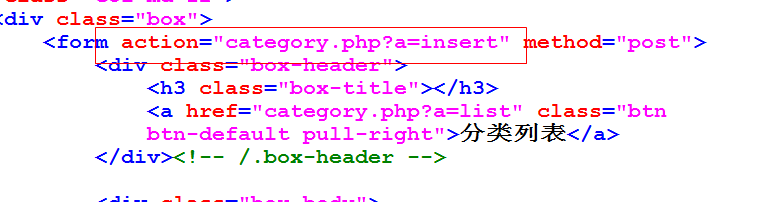
更新了category.php页面







category\_add.html 模板中，增加了一个超链接

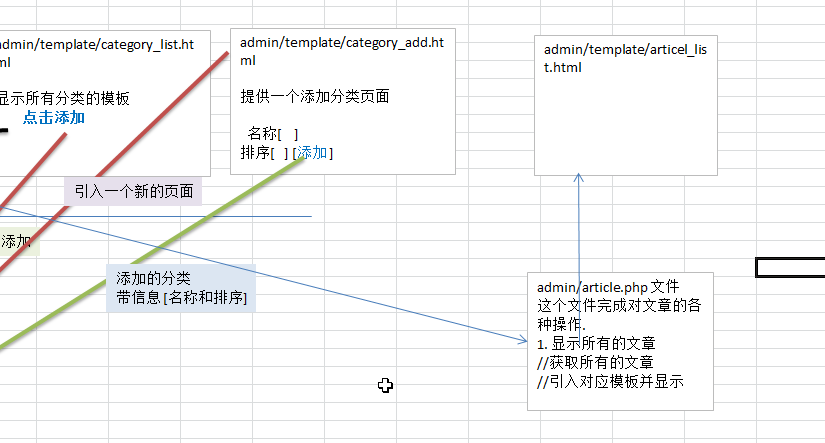


晚上的练习-写到老师这里即可

## 完成管理员可以添加文章

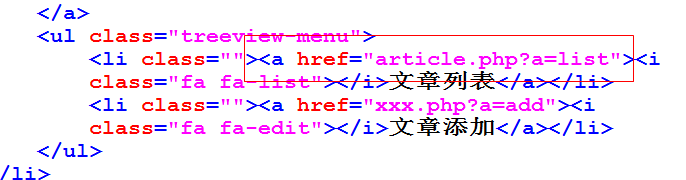
#### 完成可以在后台显示所有的文章

1. 分析完成思路和示意图

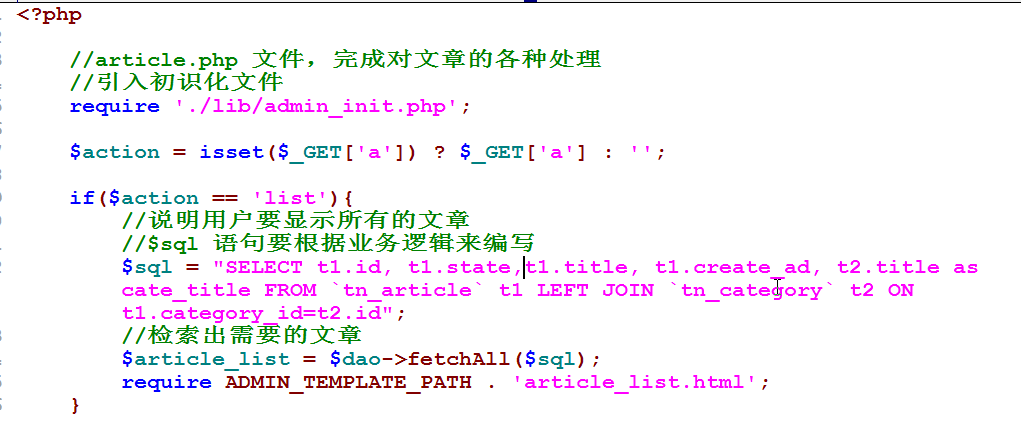


1. 代码实现

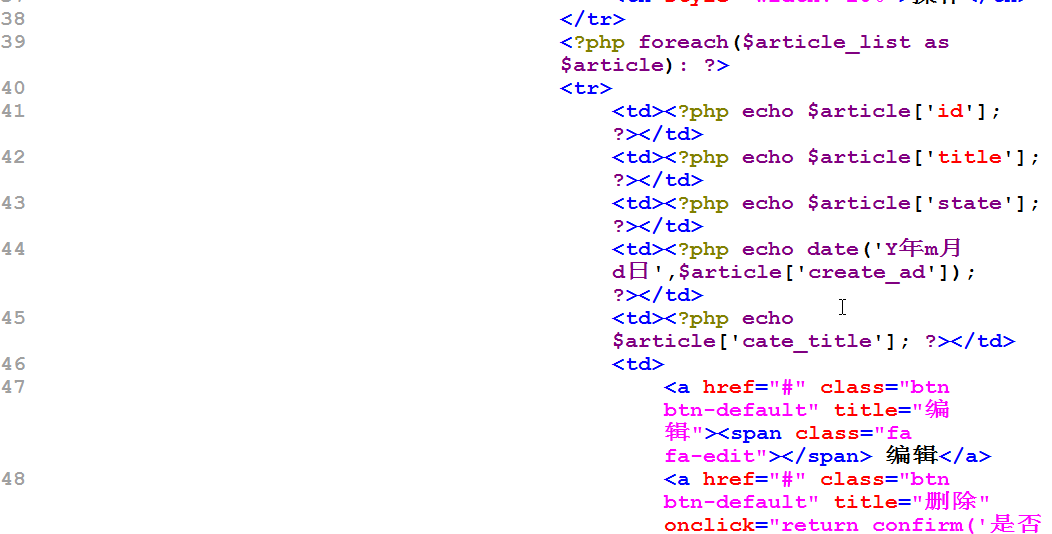
修改了left.html 模板

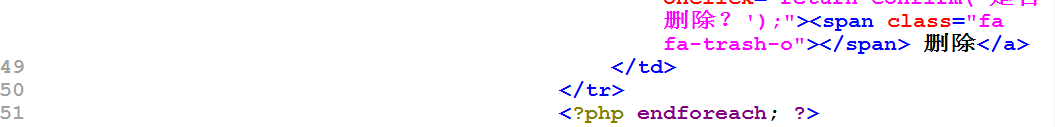


新增加了一个业务逻辑php文件 article.pp



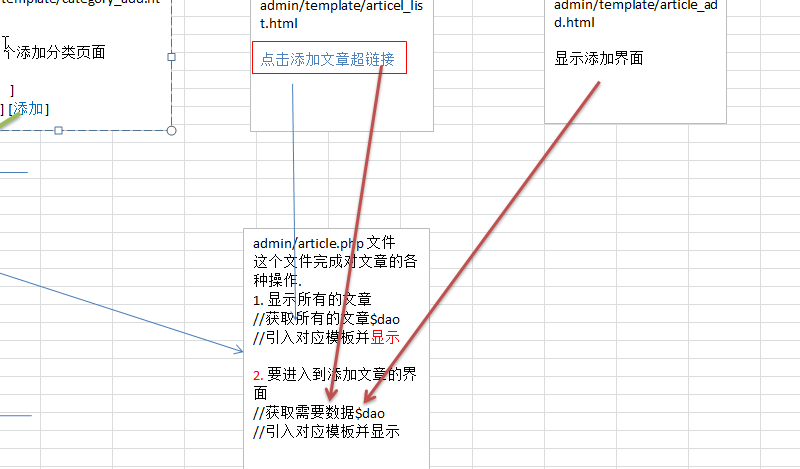
新添加了article\_list.html 模板





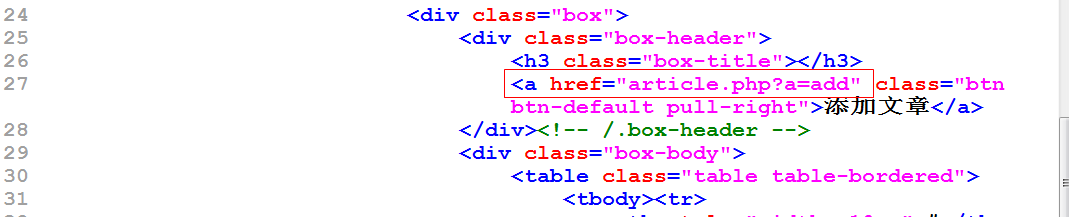
### 完成可以在后台显示添加文章的界面

1. 分析完成思路-图

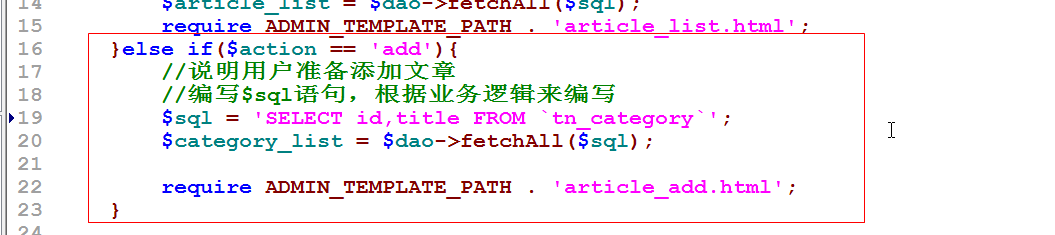


1. 完成代码

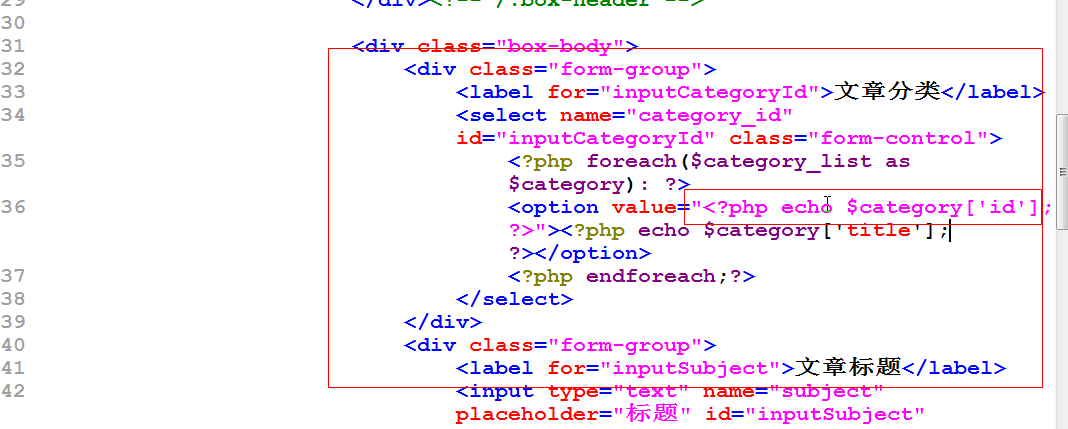
对article\_list.html 做了修改，增加了一个超链接



对article.php 文件，添加了一个新的业务逻辑

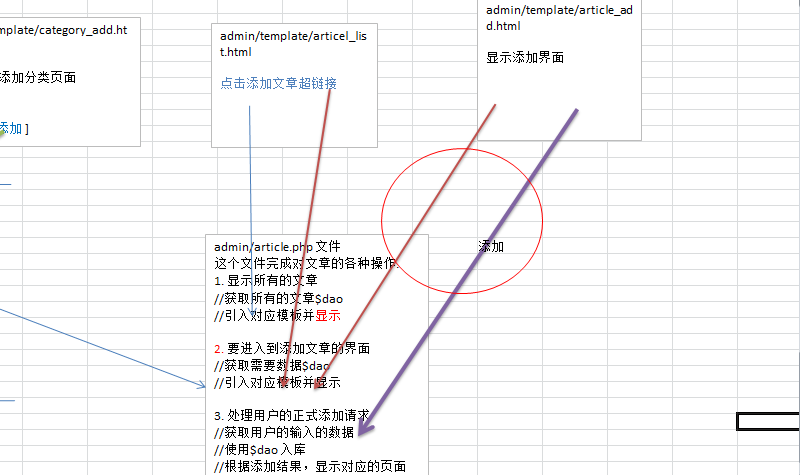


在article\_add.html模板中，我们显示了正确的页面



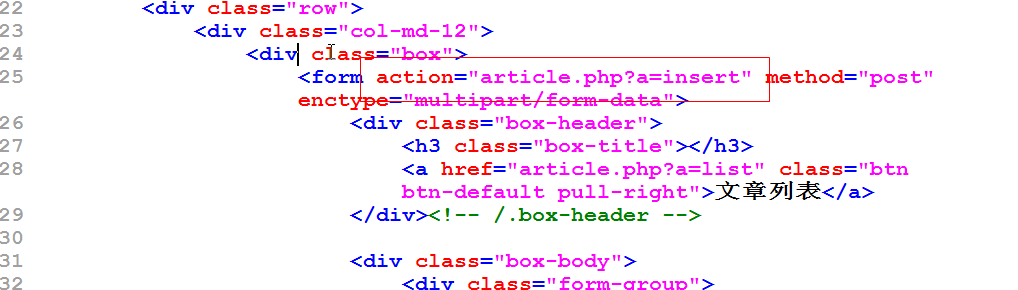
## 完成可以在后台正式添加文章到数据库中

1. 分析-示意图



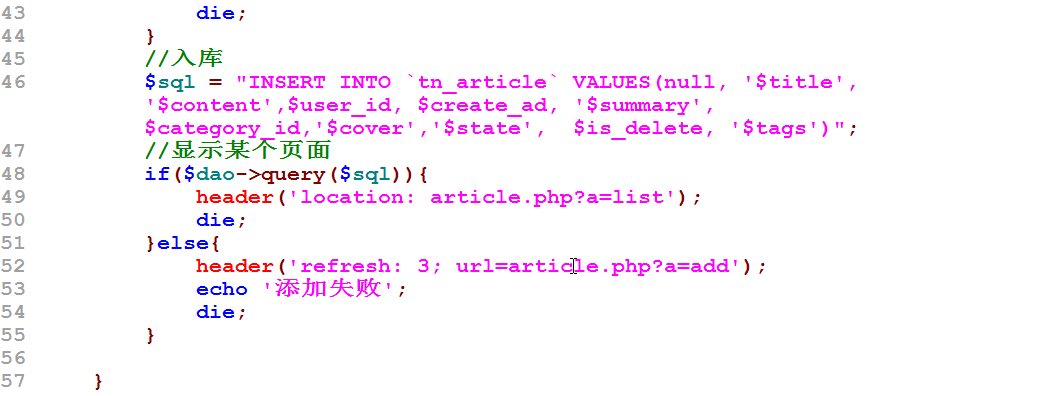
1. 代码实现

对article\_add.html 做了超链接处理



在article.php 中增加了对insert数据的逻辑代码





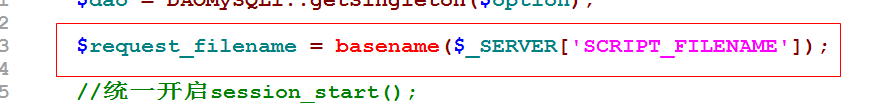
## 增加了后台管理博客时，可以正确的显示左侧left.html的操作菜单

1. 分析实现思路

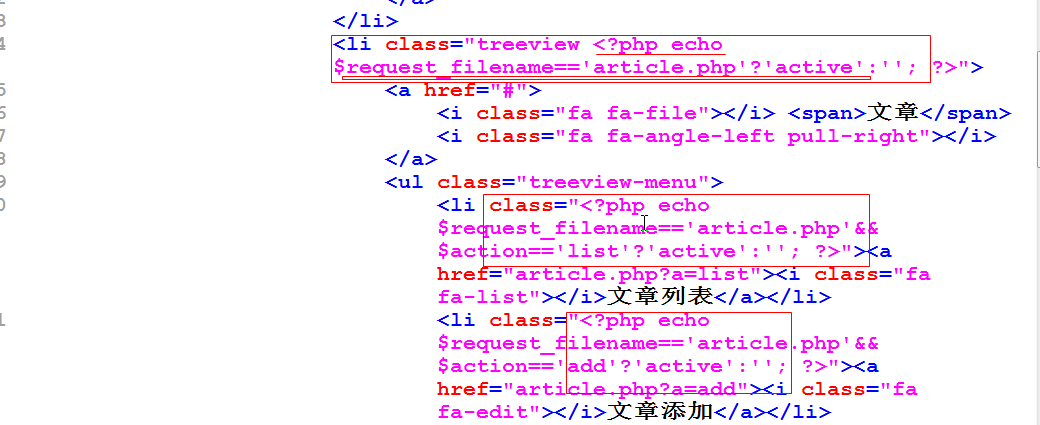
这里一定是在 left.html 页面中 出业务处理。 当用户正在操作某个功能时，css样式来显示对应的菜单.

1. 代码

修改了admin\_init.php文件



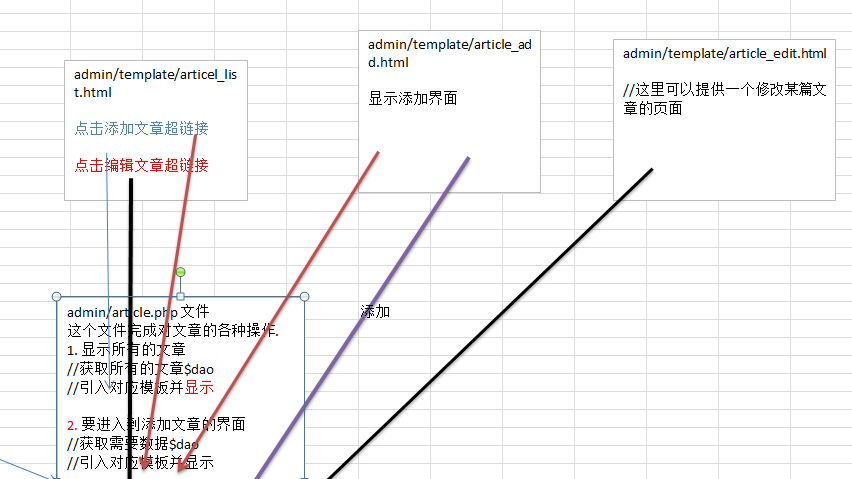
修改了admin/template/left.html 模板



还有针对分类有做了相应修改.

## 添加了分类和文章的编辑模板，要求同学们自己完成

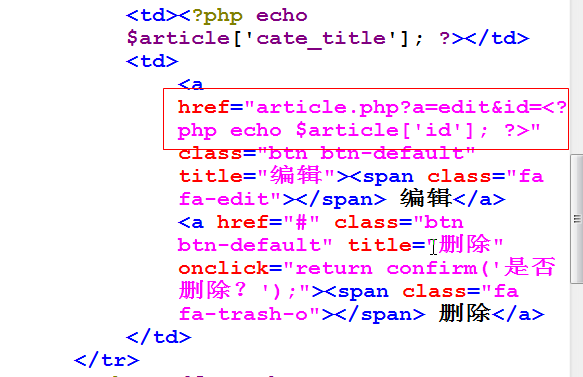
1. 分析实现思路



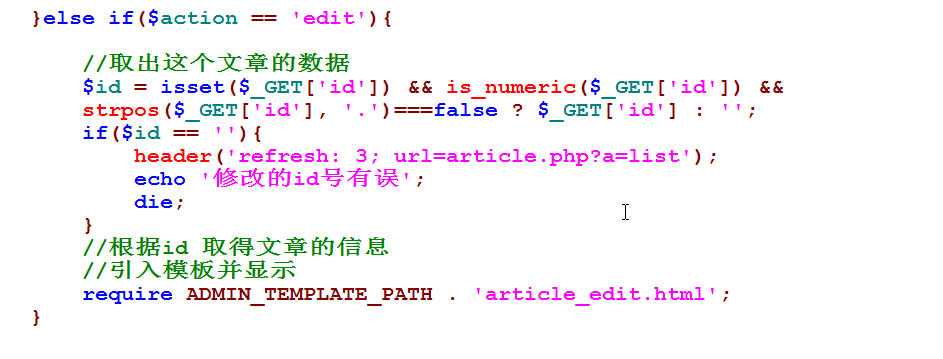


1. 代码

在article\_list.html 中增加超链接



在article.php 中增加了对应的业务逻辑代码



请同学们完成文章的修改，删除和分类的修改和删除

# 对\_GET/\_POST数据进行安全处理

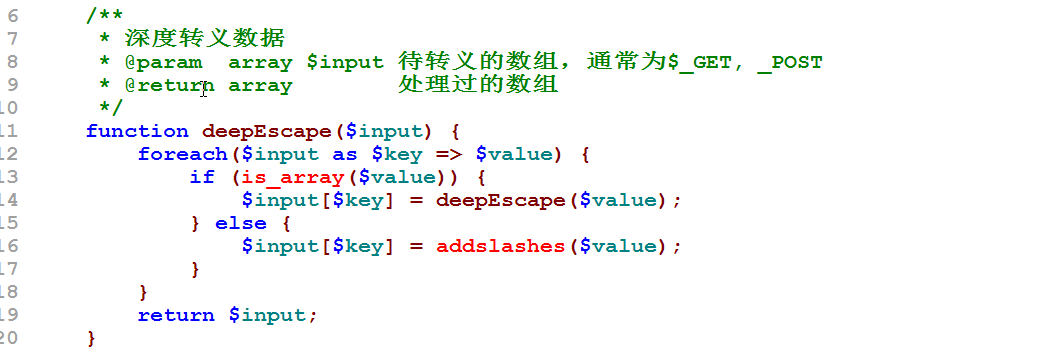
需求:

因为我们接收的数据，可能含有 一下 ' 引号，我们需要进行转义处理，重点就是对$\_GET和$\_POST的处理. ，就是对其转义 \.

解法:

即对\_GET 或者是 \_POST数据进行过滤.

先看一个深度转义的函数



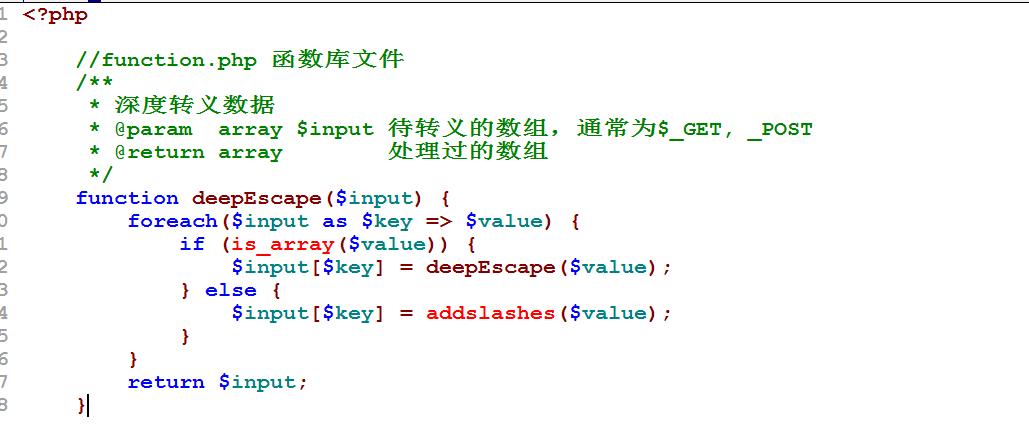
分析:

为了让所有的$\_GET和$\_POST进行转义，我们可以这样处理.在 admin\_init.php 中，我们加入这个处理, 为了很好的管理这个函数，我们单独的做一个函数库文件，在初始化化的时候引入即可.



代码实现如下:

增加一个admin/lib/function.php



在admin\_init.php 文件中引入

