后台模块

1. 品牌
2. 添加品牌：

获取试图传来的品牌图片和信息----->BrandController 控制器里面获取品牌信息，先验证信息，然后添加数据库之前先在BrandModel里面处理品牌图片--->实例化Uploda类设置大小，类型，路径之后，把品牌原图上传到指定的路径里面--->实例化Image类生成缩略图把原图覆盖掉--->把缩略图的路径存放到数据库里面

1. 删除品牌

先把数据库中的品牌信息显示出来--->BrandController 里面创建一个删除品牌的方法--->删除品牌之前必须先删除对应的品牌图片，所以BrandModel里面创建\_before\_delete方法--->\_before\_delete这个方法里面先获取brand\_logo--->去找路劲下的图片--->删除该路径下的图片--->最后删除数据库中的品牌信息

代码：

控制器--->

//删除商品品牌的方法

public function branddelt(){

$brand=D('brand');

if($brand->delete(I('id'))){

$this->success('删除品牌成功',U('brandlist'));

}else{

$this->error('删除品牌失败');

}

}

模型--->

//删除品牌之前先删除品牌图片

public function \_before\_delete($option){

$branid=$option['where']['id'];

$this->field('brand\_logo')->find($branid);

if($this->brand\_logo){

if(file\_exists($this->brand\_logo)){

unlink($this->brand\_logo);

}

}

}

1. 修改商品品牌

首先通过brandlist页面传来的id查找该id所对应的记录，然后用assign方法显示在修改页面--->修改页面里面加上隐藏input把原来logo路径和id传给brandedit方法---->修改数据之前先在BrandModel里的\_before\_update方法里面处理图片--->先删除原来的图片然后再上传新的图片--->把新图片路径加到要保存的数据里面$data

代码：

控制器：

//商品品牌修改

public function brandedit(){

$brand=D('brand');

if(IS\_POST){

if($brand->create()){

if($brand->save()){

$this->success("修改品牌成功!",U('brandlist'));

}else{

$this->error("修改品牌失败");

}

}else{

$this->error($brand->getError());

}

return ;

}

$brandinfo=$brand->find(I('id'));

$this->assign('brandinfo',$brandinfo);

$this->display();

}

模型：

//修改品牌logo

public function \_before\_update(&$data,$option){

if($\_FILES['brand\_logo']['tmp\_name']){

if(I('oldlogo')){

if(file\_exists(I('oldlogo'))){

@unlink(I('oldlogo'));

}

}

$upload = new \Think\Upload();// 实例化上传类

$upload->maxSize = 3145728 ;// 设置附件上传大小

$upload->exts = array('jpg', 'gif', 'png', 'jpeg');// 设置附件上传类型

$upload->autoSub = FALSE; //关闭自动生产子目录

$upload->rootPath = './'; // 设置附件上传根目录

$upload->savePath = './public/Upload/Brand/'; // 设置附件上传（子）目录

$info=$upload->uploadOne($\_FILES['brand\_logo']);

$log=$info['savepath'].$info['savename'];

$image = new \Think\Image();

$image->open($log);

$image->thumb(100,30)->save($log);

$data['brand\_logo']=$log;

}

}

1. 商品
2. 商品添加：

从Goodsadd.html视图里面获取数据---->把获取的数据做个验证--->添加之前先在goodmodel-->\_before\_insert方法里面把上传的商品原图处理生成大中小缩略图把原图，大中小缩略图上传到指定的目录--->把四种图片的上传路径添加到$data里面--->通过date().rand()函数来生成商品编号加到$data里面--->把onsale处理添加到$data数组里面--->把$data数组发给goodsadd方法里面，然后通过add方法来添加到数据库里面

//商品新增

public function goodsadd(){

$goods=D('goods');

if(IS\_POST){

if($goods->create()){

if($goods->add()){

$this->success('添加商品成功！',U('goodslist'));

}else{

$this->error('添加商品失败！');

}

}else{

$this->error($goods->getError());

}

return;

}

$categorys=D('category')->catetree();

$this->assign('categorys',$categorys);

$brand=D('brand')->select();

$this->assign('brands',$brand);

$this->display();

}

protected $\_validate=array(

array('goods\_name','require','商品名称不能为空！',1),

array('market\_price','require','市场价格名称不能为空！',1),

array('shop\_price','require','本店价格名称不能为空！',1),

array('goods\_name','','商品名称已存在！',1,'unique'),

);

// 生成商品缩略图

public function \_before\_insert(&$data,$option){

if($\_FILES['original']['tmp\_name']){

$upload = new \Think\Upload();// 实例化上传类

$upload->maxSize = 3145728 ;// 设置附件上传大小

$upload->exts = array('jpg', 'gif', 'png', 'jpeg');// 设置附件上传类型

$upload->autoSub = FALSE; //关闭自动生产子目录

$upload->rootPath = './'; // 设置附件上传根目录

$upload->savePath = './public/Upload/Goods/'; // 设置附件上传（子）目录

$info=$upload->uploadOne($\_FILES['original']);

$original=$info['savepath'].$info['savename'];

$max\_thumb=$info['savepath'].'max\_'.$info['savename'];

$mid\_thumb=$info['savepath'].'mid\_'.$info['savename'];

$sm\_thumb=$info['savepath'].'sm\_'.$info['savename'];

$image = new \Think\Image();

$image->open($original);

$image->thumb(362,362)->save($max\_thumb);//大缩略图

$image->thumb(222,222)->save($mid\_thumb);//中缩略图

$image->thumb(67,67)->save($sm\_thumb);//小缩略图

$data['original']=$original;

$data['sm\_thumb']=$max\_thumb;

$data['mid\_thumb']=$mid\_thumb;

$data['max\_thumb']=$sm\_thumb;

}

}

1. 商品列表：

商品表里面只存储品牌编号和分类编号，但显示商品信息的时候需要品牌名称和分类名称。所以使用TP框架的视图模型功能，通过分类id和brand\_id来把brand表，category表和good表关联起来，然后获取商品所对应的品牌名称和分类名称。创建一个GoodsViewModel类---->GoodsViewModel里获取商品表里面的基本信息，然后通过cate\_id和brand\_id来关联获取商品所对应的品牌名称和分类名称--->在goodslist方法里面实例化GoodsView类--->分页显示给goodslist视图

//品牌列表

public function goodslist(){

$goods=D('GoodsView');

$count=$goods->count();//查找品牌表的总记录数

$Page=new \Think\Page($count,2);//实例化分页类，传递总记录数和每页显示的记录数

$Page->setConfig("prev","上一页");//设置分页类的属性

$Page->setConfig("next","下一页");

$show=$Page->show();//分页显示输出

$goodsinfo=$goods->order()->limit($Page->firstRow.','.$Page->listRows)->select();

$this->assign('page',$show);

$this->assign('goodsinfo',$goodsinfo);

$this->display();

}

1. 商品的类型：

新增一个商品之前必须要先确认商品属于哪个类型。所以先往商品类型表里面添加商品类型。每一个商品类型有自己不同的属性，所以给每一个商品类型添加属于自己的属性。新增属性，修改属性和展示某类型下的属性的时候都需要对应的类型ID。删除商品类型的时候必须要先删除该类型下面的所有属性。添加商品属性的时候必须要先添加类型，再去新增属性。

class TypeModel extends Model{

protected $\_validate=array(

array('typename','require','商品类型名称不能为空！',1),

array('typename','','商品类型名称不能重复！',1,'unique'),

);

public function \_before\_delete($option){

$typeid=$option['where']['id'];

$this->execute("DELETE FROM sp\_attr where type\_id='{$typeid}'");

}

}

三，会员

会员模块是对云一个网站来时特别重要的模块之一。不同的网站有各自的会员制度方案。例如商城的会员制度就是办会员顾客每次购买商品的时候，按照本网站的会员制度把会员的消费金额转换成积分的方式记录，然后按照积分的大小把会员分不同的等级。不同等级的会员所享受的优惠方案，活动的种类和折扣率也不同。会员的积分总数达到升级的地步，通过消分把会员升级。

本网站的会员逻辑是在sp\_viplevel表里面记录会员的等级，积分下限，积分上限和会员折扣率。在sp\_vipprcie表里面记录会员价格，会员等级ID和商品ID。

添加商品的时候如果指定会员价格的话，会员购买的商品价格按指定的会员价格来算。如果没有指定会员价格的话，会员购买的商品价格按会员等级的折扣率来算。