



# 品优购电商系统开发

## 第 7 章

### 商品录入【2】

传智播客.黑马程序员

# 课程目标

目标 1: 掌握 angularJS 上传

目标 2: 完成商品录入功能



## 1.商家后台-商品录入【商品图片上传】

### 1.1 需求分析

在商品录入界面实现多图片上传

新建

删除

	颜色	图片
<input type="checkbox"/>	颜色	
<input type="checkbox"/>	黑色	
<input type="checkbox"/>	白色	

当用户点击新建按钮，弹出上传窗口

#### 上传商品图片

颜色

颜色

商品图片

选择文件

TB2516aoR4Ipu...\_400x400.jpg

上传

保存

关闭



## 1.2 后端代码

### 1.2.1 工具类

(1) pinyougou-common 工程 pom.xml 引入依赖

```
<!-- 文件上传组件 -->

<dependency>

    <groupId>org.csource.fastdfs</groupId>

    <artifactId>fastdfs</artifactId>

</dependency>

<dependency>

    <groupId>commons-fileupload</groupId>

    <artifactId>commons-fileupload</artifactId>

</dependency>
```

(2) 将“资源/fastDFS/工具类”的 FastDFSClient.java 拷贝到 pinyougou-common 工程

### 1.2.2 配置文件

(1) 将“资源/fastDFS/配置文件”文件夹中的 fdfs\_client.conf 拷贝到 pinyougou-shop-web 工程 config 文件夹

(2) 在 pinyougou-shop-web 工程 application.properties 添加配置

```
FILE_SERVER_URL=http://192.168.25.133/
```

(3) 在 pinyougou-shop-web 工程 springmvc.xml 添加配置:

```
<!-- 配置多媒体解析器 -->

<bean id="multipartResolver"
class="org.springframework.web.multipart.commons.CommonsMultipartResolver">
```



```
<property name="defaultEncoding" value="UTF-8"></property>

<!-- 设定文件上传的最大值 5MB, 5*1024*1024 -->

<property name="maxUploadSize" value="5242880"></property>

</bean>
```

### 1.2.3 控制层

在 pinyougou-shop-web 新建 UploadController.java

```
package com.pinyougou.shop.controller;

import org.springframework.beans.factory.annotation.Value;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;
import org.springframework.web.multipart.MultipartFile;
import entity.Result;
import util.FastDFSClient;

/**
 * 文件上传 Controller
 *
 * @author Administrator
 *
 */
@RestController

public class UploadController {

    @Value("${FILE_SERVER_URL}")
```



```
private String FILE_SERVER_URL;//文件服务器地址

@RequestMapping("/upload")

public Result upload( MultipartFile file){

    //1、取文件的扩展名

    String originalFilename = file.getOriginalFilename();

    String extName = originalFilename.substring(originalFilename.lastIndexOf(".")
+ 1);

    //2、创建一个 FastDFS 的客户端

    FastDFSClient fastDFSClient;

    try {

        fastDFSClient = new FastDFSClient("classpath:config/fdfs_client.conf");

        //3、执行上传处理

        String path = fastDFSClient.uploadFile(file.getBytes(), extName);

        //4、拼接返回的 url 和 ip 地址，拼装成完整的 url

        String url = FILE_SERVER_URL + path;

        return new Result(true,url);

    } catch (Exception e) {

        e.printStackTrace();

        return new Result(false, "上传失败");

    }

}
```



## 1.3 前端代码

### 1.3.1 服务层

(1) 在 pinyougou-shop-web 工程创建 uploadService.js

```
//文件上传服务层

app.service("uploadService",function($http){

    this.uploadFile=function(){

        var formData=new FormData();

        formData.append("file",file.files[0]);

        return $http({

            method:'POST',

            url:"../upload.do",

            data: formData,

            headers: {'Content-Type':undefined},

            transformRequest: angular.identity

        });

    }

});
```

angularjs 对于 post 和 get 请求默认的 Content-Type header 是 application/json。通过设置 'Content-Type': undefined，这样浏览器会帮我们吧 Content-Type 设置为 multipart/form-data。

通过设置 transformRequest: angular.identity，angularjs transformRequest function 将序列化我们的 formdata object。

(2) 将 uploadService 服务注入到 goodsController 中



//商品控制层（商家后台）

```
app.controller('goodsController' ,function($scope,$controller ,goodsService,itemCa  
tService,uploadService){
```

（3）在 goods\_edit.html 引入 js

```
<script type="text/javascript" src="../../js/base.js"> </script>  
  
<script type="text/javascript" src="../../js/service/goodsService.js"> </script>  
  
<script type="text/javascript" src="../../js/service/itemCatService.js"> </script>  
  
<script type="text/javascript" src="../../js/service/uploadService.js"> </script>  
  
<script type="text/javascript" src="../../js/controller/baseController.js"> </script>  
<script type="text/javascript" src="../../js/controller/goodsController.js"> </script>
```

## 1.3.2 上传图片

（1）goodsController 编写代码

```
/**  
 * 上传图片  
 */  
  
$scope.uploadFile=function(){  
  
    uploadService.uploadFile().success(function(response) {  
  
        if(response.success){//如果上传成功，取出 url  
  
            $scope.image_entity.url=response.message;//设置文件地址  
  
        }else{  
  
            alert(response.message);  
  
        }  
  
    }).error(function() {
```



```
        alert("上传发生错误");  
  
    });  
  
};
```

(2) 修改图片上传窗口，调用上传方法，回显上传图片

```
<div class="modal-body">  
  
    <table class="table table-bordered table-striped">  
  
        <tr>  
  
            <td>颜色</td>  
  
            <td><input class="form-control" placeholder="颜色"  
ng-model="image_entity.color"> </td>  
  
        </tr>  
  
        <tr>  
  
            <td>商品图片</td>  
  
            <td>  
  
                <table>  
  
                    <tr>  
  
                        <td>  
  
                            <input type="file" id="file" />  
  
                            <button class="btn btn-primary" type="button"  
ng-click="uploadFile()">  
  
                                上传  
  
                            </button>  
  
                        </td>  

```





```

                <td>

                </td>

            </tr>

        </table>

    </td>

</tr>

</table>

</div>

```

### (3) 修改新建按钮

```

<button type="button" class="btn btn-default" title="新建" data-target="#uploadModal"
data-toggle="modal" ng-click="image_entity={}" ><i class="fa fa-file-o"></i> 新建
</button>

```

## 1.3.3 图片列表

### (1) 在 goodsController.js 增加方法

```

$scope.entity={goods:{},goodsDesc:{itemImages:[]}};//定义页面实体结构

//添加图片列表,通过数据双向绑定机制回显图片列表

$scope.add_image_entity=function(){

    $scope.entity.goodsDesc.itemImages.push($scope.image_entity);

}

```

### (2) 修改上传窗口的保存按钮

```

<button class="btn btn-success" ng-click="add_image_entity()" data-dismiss="modal"

```



```
aria-hidden="true">保存</button>
```

### (3) 遍历图片列表

```
<tr ng-repeat="pojo in entity.goodsDesc.itemImages">

    <td>{{pojo.color}}</td>

    <td></td>

    <td><button type="button" class="btn btn-default" title="删除" ><i class="fa
fa-trash-o"></i> 删除</button></td>

</tr>
```

## 1.3.4 移除图片

在 goodsController.js 增加代码

```
//列表中移除图片

$scope.remove_image_entity=function(index){

    $scope.entity.goodsDesc.itemImages.splice(index,1);

}
```

修改列表中的删除按钮

```
<button type="button" class="btn btn-default" title="删除"
ng-click="remove_image_entity($index)"><i class="fa fa-trash-o"></i> 删除</button>
```

## 2.商家后台-商品录入【品牌选择】

### 2.1 需求分析

在用户选择商品分类后，品牌列表要根据用户所选择的分类进行更新。具体的逻辑是根据用户选择的三级分类找到对应的商品类型模板，商品类型模板中存储了品牌的列表 json 数据。



品牌

中兴

## 2.2 代码实现

- (1) 在 pinyougou-shop-web 工程创建 TypeTemplateController （可从运营商后台拷贝）
- (2) 在 pinyougou-shop-web 工程创建 type\_templateService.js （可从运营商后台拷贝）
- (3) 在 goodsController 引入 type\_templateService 并新增代码

```
//模板 ID 选择后 更新品牌列表

$scope.$watch('entity.goods.typeTemplateId', function(newValue, oldValue) {

    typeTemplateService.findOne(newValue).success(

        function(response){

            $scope.typeTemplate=response;//获取类型模板

            $scope.typeTemplate.brandIds=
JSON.parse( $scope.typeTemplate.brandIds);//品牌列表

        }

    );

});
```

在页面 goods\_edit.html 引入 js

```
<script type="text/javascript" src="../js/service/typeTemplateService.js"> </script>
```

添加品牌选择框

```
<select class="form-control" ng-model="entity.goods.brandId" ng-options="item.id as
item.text for item in typeTemplate.brandIds"></select>
```



## 3.商家后台-商品录入【扩展属性】

### 3.1 需求分析

在商品录入实现扩展属性的录入

商品基本信息	商品图片	扩展属性
安卓	安卓	
网络类型	网络类型	
手机产地	手机产地	

保存

### 3.2 代码实现

修改 goodsController.js ，在用户更新模板 ID 时，读取模板中的扩展属性赋给商品的扩展属性。

```
//模板 ID 选择后 更新模板对象

$scope.$watch('entity.goods.typeTemplateId', function(newValue, oldValue) {

    typeTemplateService.findOne(newValue).success(

        function(response){

            $scope.typeTemplate=response;//获取类型模板

            $scope.typeTemplate.brandIds=
JSON.parse( $scope.typeTemplate.brandIds);//品牌列表

            $scope.entity.goodsDesc.customAttributeItems=JSON.parse( $scope.typeTemplate.custom
AttributeItems);//扩展属性

        }

    );

});
```



修改 goods\_edit.html

```
<!--扩展属性-->

<div class="tab-pane" id="customAttribute">

    <div class="row data-type">

        <div ng-repeat="pojo in entity.goodsDesc.customAttributeItems">

            <div class="col-md-2 title">{{pojo.text}}</div>

            <div class="col-md-10 data">

                <input class="form-control" ng-model="pojo.value" placeholder="{{pojo.text}}">

            </div>

        </div>

    </div>

</div>
```

## 4.商家后台-商品录入【规格选择】

### 4.1 需求分析

显示规格及选项列表（复选框）如下图，并保存用户选择的结果

网络制式	<input type="checkbox"/> 移动3G <input type="checkbox"/> 移动4G <input type="checkbox"/> 联通3G <input type="checkbox"/> 联通4G
屏幕尺寸	<input type="checkbox"/> 6寸 <input type="checkbox"/> 5.5寸 <input type="checkbox"/> 5寸 <input type="checkbox"/> 4.5寸

### 4.2 代码实现

#### 4.2.1 显示规格选项列表

由于我们的模板中只记录了规格名称，而我们除了显示规格名称还是显示规格下的规格选项，所以我们需要在后端扩充方法。



(1) 在 pinyougou-sellergoods-interface 的 TypeTemplateService.java 新增方法定义

```
/**
 * 返回规格列表
 * @return
 */

public List<Map> findSpecList(Long id);
```

(2) 在 pinyougou-sellergoods-service 的 TypeTemplateServiceImpl.java 新增方法

```
@Autowired

private TbSpecificationOptionMapper specificationOptionMapper;

@Override

public List<Map> findSpecList(Long id) {

    // TODO Auto-generated method stub

    TbTypeTemplate typeTemplate = typeTemplateMapper.selectByPrimaryKey(id);

    String specIds = typeTemplate.getSpecIds();

    try {

        List<Map> list = JSON.parse(specIds, List.class);

        for(Map map:list){

            //根据规格 ID 查询规格选项列表

            TbSpecificationOptionExample example=new
TbSpecificationOptionExample();

            Criteria criteria = example.createCriteria();

            criteria.andSpecIdEqualTo((Long)map.get("id"));
```



```

        List<TbSpecificationOption> options =
specificationOptionMapper.selectByExample(example);

        map.put("options", options);

    }

    return list;

} catch (ParseException e) {

    // TODO Auto-generated catch block

    e.printStackTrace();

    return null;

}

}

```

(3) 在 pinyougou-shop-web 的 TypeTemplateController.java 新增方法

```

@RequestMapping("/findSpecList")

public List<Map> findSpecList(Long id){

    return typeTemplateService.findSpecList(id);

}

```

测试后端代码：

⏪ ⏩ ↻ localhost:9102/typeTemplate/findSpecList.do?id=35

```

[{"id":27,"text":"网络制式","options":[{"id":98,"optionName":"移动3G","orders":1,"specId":27}, {"id":99,"optionName":"移动4G","orders":2,"specId":27}, {"id":100,"optionName":"联通3G","orders":3,"specId":27}, {"id":101,"optionName":"联通4G","orders":4,"specId":27}], {"id":28,"text":"屏幕尺寸","options": []}]

```

(4) 前端代码：修改 pinyougou-shop-web 的 typeTemplateService.js

```

//查询规格列表

this.findSpecList=function(id){

    return $http.get('../typeTemplate/findSpecList.do?id='+id);

}

```



```
}
```

#### (5) 修改 pinyougou-shop-web 的 goodsController.js

```
//模板 ID 选择后 更新模板对象

$scope.$watch('entity.goods.typeTemplateId', function(newValue, oldValue) {

    typeTemplateService.findOne(newValue).success(

        function(response){

            $scope.typeTemplate=response;//获取类型模板

            $scope.typeTemplate.brandIds=
JSON.parse( $scope.typeTemplate.brandIds);//品牌列表

$scope.entity.goodsDesc.customAttributeItems=JSON.parse( $scope.typeTemplate.custom
AttributeItems);//扩展属性

        }

    );

    //查询规格列表

    typeTemplateService.findSpecList(newValue).success(

        function(response){

            $scope.specList=response;

        }

    );

});

});
```

### 4.2.2 保存选中规格选项

我们需要将用户选中的选项保存在 tb\_goods\_desc 表的 specification\_items 字段中，定义 json 格式如下：

```
[{"attributeName":"规格名称","attributeValue":["规格选项 1","规格选项 2".... ] }, .... ]
```





(1) 在 goodsController.js 增加代码

```
$scope.entity = {goods:{},goodsDesc:{itemImages:[],specificationItems:[]}};

//更新规格选项

$scope.updateSpecAttribute=function($event,name,value){

    //查询在数组中是否存在 name

    var specList= $scope.entity.goodsDesc.specificationItems;

    for(var i=0;i<specList.length;i++){

        if(specList[i].attributeName==name){//如果 name 存在

            if($event.target.checked){//如果是选中添加值

                specList[i].attributeValue.push(value);

            }else{

                //否则移除值

                specList[i].attributeValue.splice(

                    specList[i].attributeValue.indexOf(value),1);

            }

            return ;//return 必须在这个位置

        }

    }

    //如果 name 不存在,新增一行

    specList.push({attributeName:name,attributeValue:[value]} );

}
```

(2) 在 goods\_edit.html 调用方法



```
<div ng-repeat="pojo in specList">

    <div class="col-md-2 title">{{pojo.text}}</div>

    <div class="col-md-10 data">

        <span ng-repeat="option in pojo.options">

            <input type="checkbox"

ng-click="updateSpecAttribute($event,pojo.text,option.optionName)">{{option.optionName}}

        </span>
    </div>

</div>
```

为了方便测试，我们可以在页面上某个区域临时添加表达式，以便观测测试结果

```
{{entity.goodsDesc.specificationItems}}
```

## 5.商家后台-商品录入【SKU 商品信息】

### 5.1 需求分析

基于上一步我们完成的规格选择，根据选择的规格录入商品的 SKU 信息，当用户选择相应的规格，下面的 SKU 列表就会自动生成，如下图：

屏幕尺寸	<input checked="" type="checkbox"/> 4.0 <input checked="" type="checkbox"/> 4.5 <input type="checkbox"/> 5.0 <input type="checkbox"/> 5.5 <input type="checkbox"/> 6.0		
网络制式	<input checked="" type="checkbox"/> GSM ( 移动/联通2G ) <input checked="" type="checkbox"/> 电信2G <input type="checkbox"/> 联通3G <input type="checkbox"/> 移动3G <input type="checkbox"/> 电信3G <input type="checkbox"/> 电信4G <input type="checkbox"/> 移动4G <input type="checkbox"/> 联通4G		
服装尺码	<input type="checkbox"/> 小号 <input type="checkbox"/> 大号		
网络制式	屏幕尺寸	价格	库存
GSM ( 移动/联通2G )	4.0	<input type="text" value="0"/>	<input type="text" value="99999"/>
GSM ( 移动/联通2G )	4.5	<input type="text" value="0"/>	<input type="text" value="99999"/>
电信2G	4.0	<input type="text" value="0"/>	<input type="text" value="99999"/>
电信2G	4.5	<input type="text" value="0"/>	<input type="text" value="99999"/>

实现思路：实现思路：





```
list=addColumn(list,

    $scope.entity.goodsDesc.specificationItems[i].attributeName,

    $scope.entity.goodsDesc.specificationItems[i].attributeValue);

}

$scope.entity.skuList=list;//商品 SKU 列表

}

//添加列信息

addColumn=function(list,columnName,columnValues){

    var newList=[];//新集合

    for(var i=0;i<list.length;i++){

        var oldRow=list[i];//取出原有的行

        //根据选项产生新行

        for(var j=0;j<columnValues.length;j++){

            var newRow= JSON.parse( JSON.stringify(oldRow));//深克隆操作

            newRow.spec[columnName]=columnValues[j];

            newList.push(newRow);

        }

    }

    return newList;

}
```

(2) 在更新规格属性后调用生成 SKU 列表的方法

```
<input type="checkbox" type="checkbox"
ng-click="updateSpecAttribute($event,pojo.text,option.optionName);createSKUTable()"
>{{option.optionName}}
```



(3) goods\_edit.html 页面上绑定 SKU 列表

```
<table class="table table-bordered table-striped table-hover dataTable">

  <thead>

    <tr>

      <th class="sorting" ng-repeat="item in
entity.goodsDesc.specificationItems">{{item.attributeName}}</th>

      <th class="sorting">价格</th>

      <th class="sorting">库存</th>

      <th class="sorting">是否启用</th>

      <th class="sorting">是否默认</th>

    </tr>

  </thead>

  <tbody>

    <tr ng-repeat="pojo in entity.skuList">

      <td ng-repeat="item in entity.goodsDesc.specificationItems">

        {{pojo.spec[item.attributeName]}}

      </td>

      <td>

        <input class="form-control" ng-model="pojo.price" placeholder="价格
">

      </td>

      <td>

        <input class="form-control" ng-model="pojo.stockCount" placeholder="库存
数量">
```



```
</td>

<td>

    <input type="checkbox" ng-model="pojo.status" ng-true-value="1"
ng-false-value="0" >

</td>

<td>

    <input type="checkbox" ng-model="pojo.isDefault" ng-true-value="1"
ng-false-value="0">

</td>

</tr>

</tbody>

</table>
```

## 5.3 后端代码

(1) 修改 Goods 类，增加属性

```
private List skuList;//sku 商品列表

public List getSkuList() {

    return skuList;

}

public void setSkuList(List skuList) {

    this.skuList = skuList;

}
```

(2) 在 GoodsServiceImpl 添加对 itemMapper

```
@Autowired
```



```
private TbItemMapper itemMapper;
```

(3) 修改 GoodsServiceImpl 的 add 方法，增加代码，实现对 SKU 商品信息的保存

```
//向 item 表添加 SKU 商品信息

List<Map> skuList = goods.getSkulist();

for( Map map:skuList ){

    //提取规格描述, 返回字符串

    String specStr=" ";

    Map<String,Object> specMap = (Map) map.get("spec");//提取规格信息

    for(String specKey: specMap.keySet() ){

        specStr+=" "+specMap.get(specKey);

    }

    //构建 SKU 信息

    TbItem item=new TbItem();

    item.setTitle(goods.getGoods().getGoodsName()+specStr);//商品 KPU+规格描述串作为 SKU 名称

    item.setPrice( new BigDecimal ( (String)map.get("price") ) );//价格

    item.setStockCount( (Integer)map.get("stockCount"));//库存量

    item.setStatus((String)map.get("status"));//状态

    item.setIsDefault((String)map.get("isDefault"));//是否默认

    item.setGoodsId(goods.getGoods().getId());//商品 SPU 编号

    item.setSellerId(goods.getGoods().getSellerId());//商家编号

    itemMapper.insert(item);

}
```



## 6.商家后台-商品录入【是否启用规格】

### 6.1 需求分析

在规格面板添加是否启用规格，当用户没有选择该项，将原来的规格面板和 SKU 列表隐藏，用户保存商品后只生成一个 SKU.

商品基本信息 商品图片 扩展属性 查询属性 规格

是否启用规格 ☐

保存

### 6.2 前端代码

goods\_add.html 添加复选框

```
<div class="row data-type">

    <div class="col-md-2 title">是否启用规格</div>

    <div class="col-md-10 data">

        <input type="checkbox" ng-model="entity.goods.isEnableSpec"
ng-true-value="1" ng-false-value="0">

    </div>

</div>
```

将原来的规格选项面板和 SKU 面板用 div 包起来

```
<div ng-if="entity.goods.isEnableSpec==1">

.....

</div>
```





## 6.3 后端代码

修改 GoodsServiceImpl 的 add 方法

```
if("1".equals(goods.getGoods().getIsEnableSpec())){//如果启用规格，需要循环添加 SKU

    //原来的 SKU 存储逻辑代码

}else{//如果不启用规格，则添加一个 SKU

    //构建 SKU 信息

    TbItem item=new TbItem();

    item.setTitle(goods.getGoods().getGoodsName());//商品 KPU+规格描述串作为
SKU 名称

    item.setPrice( goods.getGoods().getPrice() );//价格

    item.setStatus("1");//状态

    item.setIsDefault("1");//是否默认

    item.setGoodsId(goods.getGoods().getId());//商品 SPU 编号

    item.setSellerId(goods.getGoods().getSellerId());//商家编号

    itemMapper.insert(item);

}
```