**品优购电商系统开发**

**第4章**

**规格及模板管理**

传智播客.黑马程序员

# 1.前端分层开发

## 1.1 需求分析

我们在上次课学习了angularJS并完成的品牌管理的增删改查功能。但是我们看代码，JS和html都放在一起，并不利于我们后期的维护。我们可以在前端代码中也运用MVC的设计模式，将代码进行分离，提高程序的可维护性。

## 1.2 自定义服务

在 AngularJS 中，服务是一个函数或对象，可在你的 AngularJS 应用中使用。我们在上次课中使用了内置服务$http .其实我们也可以自己来定义服务，而服务会封装一些操作。我们在不同的控制器中可以调用同一个服务，这样服务的代码将会被重用。

我们现在就修改一下我们的品牌管理代码，使用自定义服务。

## 1.3代码分离

我们刚才已经将与后端交互的部分放入自定义服务，目的是不同的控制层都可以重复调用服务层方法。所以我们还需要将代码分离出来，以便调用。

**修改页面**

去掉brand.html原来的JS代码，引入JS

|  |
| --- |
| <script type=*"text/javascript"* src=*"../plugins/angularjs/angular.min.js"*> </script>  <!-- 分页组件开始 -->  <script src=*"../plugins/angularjs/pagination.js"*></script>  <link rel=*"stylesheet"* href=*"../plugins/angularjs/pagination.css"*>  <!-- 分页组件结束 -->  <script type=*"text/javascript"* src=*"../js/base\_pagination.js"*> </script>  <script type=*"text/javascript"* src=*"../js/controller/baseController.js"*> </script>  <script type=*"text/javascript"* src=*"../js/controller/brandController.js"*> </script> |

# 2.控制器继承

## 2.1需求分析

有些功能是每个页面都有可能用到的，比如分页，复选等等，如果我们再开发另一个功能，还需要重复编写。怎么能让这些通用的功能只写一次呢？我们通过继承的方式来实现。

## 2.2前端代码

### **2.2.1 父控制器**

在manager的js/controller目录下的baseController.js

|  |
| --- |
| //基本控制层  app.controller('baseController' ,**function**($scope){  //重新加载列表 数据  $scope.reloadList=**function**(){  //切换页码  $scope.search( $scope.paginationConf.currentPage, $scope.paginationConf.itemsPerPage);  }  //分页控件配置  $scope.paginationConf = {  currentPage: 1,  totalItems: 0,  itemsPerPage: 10,  perPageOptions: [10, 20, 30, 40, 50],  onChange: **function**(){  $scope.reloadList();//重新加载  }  };  $scope.selectIds=[];//选中的ID集合  //更新复选  $scope.updateSelection = **function**($event, id) {  **if**($event.target.checked){//如果是被选中,则增加到数组  $scope.selectIds.push( id);  }**else**{  **var** idx = $scope.selectIds.indexOf(id);  $scope.selectIds.splice(idx, 1);//删除  }  }  }); |

### **2.2.2 品牌控制器层**

brandController.js

|  |
| --- |
| //品牌控制层  app.controller('brandController' ,**function**($scope,$controller,brandService){  $controller('baseController',{$scope:$scope});//继承  //读取列表数据绑定到表单中  $scope.findAll=**function**(){  brandService.findAll().success(  **function**(response){  $scope.list=response;  }  );  }  //其它方法省略........  }); |

$controller也是angular提供的一个服务，可以实现伪继承，实际上就是与BaseController共享$scope

### **2.2.2 品牌业务层**

|  |
| --- |
| // 定义服务层:  app.service("brandService",**function**($http){  **this**.findAll = **function**(){  **return** $http.get("../brand/findAll.do");  }    **this**.findByPage = **function**(page,rows){  **return** $http.get("../brand/findByPage.do?page="+page+"&rows="+rows);  }    **this**.save = **function**(entity){  **return** $http.post("../brand/save.do",entity);  }    **this**.update=**function**(entity){  **return** $http.post("../brand/update.do",entity);  }    **this**.findById=**function**(id){  **return** $http.get("../brand/findById.do?id="+id);  }    **this**.dele = **function**(ids){  **return** $http.get("../brand/delete.do?ids="+ids);  }    **this**.search = **function**(page,rows,searchEntity){  **return** $http.post("../brand/search.do?page="+page+"&rows="+rows,searchEntity);  }  }); |

# 3.规格管理

## 3.1需求及表结构分析

### **3.1.1 需求**

实现规格管理功能

### **3.1.2 表结构**

**tb\_specification 规格表**

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 类型 | 长度 | 含义 |
| Id | Bigint |  | 主键 |
| spec\_name | Varchar | 255 | 规格名称 |

**tb\_specification\_option 规格选项表**

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 类型 | 长度 | 含义 |
| id | Bigint |  | 主键 |
| option\_name | Varchar | 200 | 规格选项名称 |
| spec\_id | Bigint | 30 | 规格ID |
| orders | Int | 11 | 排序 |

## 3.2规格列表



### **3.2.1 引入JS**

修改pinyougou-web-manager工程的specification.html

|  |
| --- |
| <script type=*"text/javascript"* src=*"../plugins/angularjs/angular.min.js"*> </script>  <script src=*"../plugins/angularjs/pagination.js"*></script>  <link rel=*"stylesheet"* href=*"../plugins/angularjs/pagination.css"*>  <script type=*"text/javascript"* src=*"../js/base\_pagination.js"*> </script>  <script type=*"text/javascript"* src=*"../js/service/specificationService.js"*> </script>  <script type=*"text/javascript"* src=*"../js/controller/baseController.js"*></script>  <script type=*"text/javascript"* src=*"../js/controller/specificationController.js"*> </script> |

### **3.2.2 放置分页组件**

|  |
| --- |
| <!-- 分页 -->  <tm-pagination conf=*"paginationConf"*></tm-pagination> |

### **3.2.3 指令与表达式**

在body元素指定模块名和控制器名

|  |
| --- |
| <body class=*"hold-transition skin-red sidebar-mini"*  ng-app=*"pinyougou"* ng-controller=*"specificationController"* > |

循环表格行

|  |
| --- |
| <tr ng-repeat=*"entity in list"*>  <td><input type=*"checkbox"* ></td>  <td>{{entity.id}}</td>  <td>{{entity.specName}}</td>  <td class=*"text-center"*>  <button type=*"button"* class=*"btn bg-olive btn-xs"* data-toggle=*"modal"* data-target=*"#editModal"*>修改</button>  </td>  </tr> |

### **3.2.4 sellerGoods中SpecificationServiceImpl 中添加**

|  |
| --- |
| // 返回分页对象 查询条件  **public** PageResult search(Integer pageNum, Integer pageSize, Specification specification) {  SpecificationQuery specificationQuery = **new** SpecificationQuery();  // 判断查询条件  **if** (**null** != specification.getSpecName()) {  specificationQuery.createCriteria().andSpecNameLike("%" + specification.getSpecName() + "%");  }  // 使用的是angularjs的分页对象 当前页肯定有值 每页数 显示也有值 不可能是Null  PageHelper.*startPage*(pageNum, pageSize);  Page<Specification> page = (Page<Specification>) specificationDao.selectByExample(specificationQuery);  **return** **new** PageResult(page.getTotal(), page.getResult());  } |

### **3.2.4 manager中SpecificationController 中添加**

|  |
| --- |
| //查询规格列表 分页 + 查询条件  @RequestMapping(value = "/specification/search.do")  **public** PageResult search(Integer page,Integer rows,@RequestBody Specification specification){  **return** specificationService.search(page, rows, specification);  } |

## 3.3新增规格



### **3.3.1 新增行的实现**

修改specificationController.js 新增以下代码

|  |
| --- |
| //新增选项行  $scope.addTableRow=**function**(){  $scope.entity.specificationOptionList.push({});  } |

specification.html “新建选项”按钮

|  |
| --- |
| <button type=*"button"* class=*"btn btn-default"* title=*"新建"* ng-click=*"addTableRow()"*><i class=*"fa fa-file-o"*></i> 新建</button> |

循环列表行，绑定表格内的编辑框

|  |
| --- |
| <tr ng-repeat=*"pojo in entity.specificationOptionList"*>  <td>  <input ng-model=*"pojo.optionName"* class=*"form-control"* placeholder=*"规格选项"*/>  </td>  <td>  <input ng-model=*"pojo.orders"* class=*"form-control"* placeholder=*"排序"*/>  </td>  <td>  <button type=*"button"* class=*"btn btn-default"* title=*"删除"* ng-click=*"deleTableRow($index)"*><i class=*"fa fa-file-o"*></i> 删除</button>  </td>  </tr> |

注意：要修改specification.html “新建”按钮，弹出窗口时对entity进行初始化，否则向集合添加数据时会报错！

|  |
| --- |
| <button type=*"button"* class=*"btn btn-default"* title=*"新建"* data-toggle=*"modal"* data-target=*"#editModal"* ng-click=*"entity={'specificationOptionList':[]}"*><i class=*"fa fa-file-o"*></i> 新建</button> |

### **3.3.2 删除行的实现**

实现思路：在每一行将索引值传递给集合，在集合中删除。

修改specificationController.js 新增以下代码

|  |
| --- |
| //批量选项删除  $scope.deleTableRow=**function**(index){  $scope.entity.specificationOptionList.splice(index,1);//删除  } |

修改每行的删除按钮

|  |
| --- |
| <button type=*"button"* class=*"btn btn-default"* title=*"删除"* ng-click=*"deleTableRow($index)"*><i class=*"fa fa-file-o"*></i> 删除</button> |

$index 用于获取ng-repeat指令循环中的索引。

### **3.3.3 提交保存**

实现思路：我们将规格和规格选项数据合并成一个对象来传递，这时我们需要用一个对象将这两个对象组合起来。在业务逻辑中，得到组合对象中的规格和规格选项列表，插入规格返回规格ID，然后循环插入规格选项。

1. 我们要增加规格选项，必须要知道新增规格的ID, 所以我们在修改pinyougou-dao 的TbSpecificationMapper.xml ，在insert节点后添加如下配置

|  |
| --- |
| <insert id=*"insert"* parameterType=*"com.pinyougou.pojo.TbSpecification"* >  <selectKey resultType=*"java.lang.Long"* order=*"AFTER"* keyProperty=*"id"*>  SELECT LAST\_INSERT\_ID() AS id  </selectKey>  insert into tb\_specification (id, spec\_name)  values (#{id,jdbcType=BIGINT}, #{specName,jdbcType=VARCHAR})  </insert> |

（2）在pinyougou-pojo 建立com.pinyougou.pojogroup包，包下建立Specification类

|  |
| --- |
| /\*\*  \* 规格组合实体类  \* **@author** lx  \*  \*/  **public** **class** SpecificationVo **implements** Serializable{  /\*\*  \*  \*/  **private** **static** **final** **long** ***serialVersionUID*** = 1L;  //规格对象  **private** Specification specification;  //规格属性对象  **private** List<SpecificationOption> specificationOptionList; |

1. 修改sellergoods的SpecificationServiceImpl.java

|  |
| --- |
| /\*\*  \* 增加  \*/  @Override  **public** **void** add(SpecificationVo vo) {  //保存规格表 并返回规格的ID  specificationDao.insertSelective(vo.getSpecification());  //保存规格属性表 多个  List<SpecificationOption> specificationOptionList = vo.getSpecificationOptionList();  **for** (SpecificationOption specificationOption : specificationOptionList) {  //设置规格表的ID 规格属性表的外键  specificationOption.setSpecId(vo.getSpecification().getId());    specificationOptionDao.insertSelective(specificationOption);  }    } |

（5）修改manager的SpecificationController.java

|  |
| --- |
| /\*\*  \* 增加  \* **@param specification**  \* **@return**  \*/  @RequestMapping("/add")  **public** Result add(@RequestBody SpecificationVo vo){  **try** {  specificationService.add(vo);  **return** **new** Result(**true**, "增加成功");  } **catch** (Exception e) {  e.printStackTrace();  **return** **new** Result(**false**, "增加失败");  }  } |

1. 修改页面specification.html

绑定规格名称

|  |
| --- |
| <table class=*"table table-bordered table-striped"* width=*"800px"*>  <tr>  <td>规格名称</td>  <td>  <input ng-model=*"entity.specification.specName"* class=*"form-control"* placeholder=*"规格名称"* >  </td>  </tr>  </table> |

绑定保存按钮事件

|  |
| --- |
| <button class=*"btn btn-success"* data-dismiss=*"modal"* aria-hidden=*"true"* ng-click=*"save()"*>保存</button> |

## 3.4修改规格



### **3.4.1 获取规格数据**

实现思路：通过规格ID，到后端查询规格和规格选项列表，然后通过组合实体类返回结果

（1）修改pinyougou-sellergoods-service的SpecificationServiceImpl.java

|  |
| --- |
| ////通过规格ID 查询规格表 及规格属性表结果集  **public** SpecificationVo findOne(Long id){  SpecificationVo vo = **new** SpecificationVo();  vo.setSpecification(specificationDao.selectByPrimaryKey(id));  SpecificationOptionQuery query = **new** SpecificationOptionQuery();  query.createCriteria().andSpecIdEqualTo(id);  vo.setSpecificationOptionList(specificationOptionDao.selectByExample(query));  **return** vo;  } |

（2）修改pinyougou-manager-web的SpecificationController.java

|  |
| --- |
| //通过规格ID 查询规格表 及规格属性表结果集  @RequestMapping(value = "/findOne.do")  **public** SpecificationVo findOne(Long id){  **return** specificationService.findOne(id);  } |

（3）修改页面specification.html 中列表的修改按钮

|  |
| --- |
| <button type=*"button"* class=*"btn bg-olive btn-xs"* data-toggle=*"modal"* data-target=*"#editModal"* ng-click=*"findOne(entity.id)"*>修改</button> |

### **3.4.2 保存修改结果**

（1）修改pinyougou-sellergoods-service的SpecificationServiceImpl.java

|  |
| --- |
| //修改规格表及规格属性表  **public** **void** update(SpecificationVo vo){  //修改规格表  specificationDao.updateByPrimaryKeySelective(vo.getSpecification());  //删除规格属性表  SpecificationOptionQuery query = **new** SpecificationOptionQuery();  query.createCriteria().andSpecIdEqualTo(vo.getSpecification().getId());  specificationOptionDao.deleteByExample(query);  //再保存规格属性表  List<SpecificationOption> specificationOptionList = vo.getSpecificationOptionList();  **for** (SpecificationOption specificationOption : specificationOptionList) {  //设置规格表的ID 规格属性表的外键  specificationOption.setSpecId(vo.getSpecification().getId());  specificationOptionDao.insertSelective(specificationOption);  }    } |

（2）修改pinyougou-manager-web的SpecificationController.java

|  |
| --- |
| //修改  @RequestMapping(value = "/update")  **public** Result update(@RequestBody SpecificationVo vo){  **try** {    specificationService.update(vo);  **return** **new** Result(**true**,"修改成功");  } **catch** (Exception e) {  // **TODO**: handle exception  e.getStackTrace();  **return** **new** Result(**false**,"修改失败");  }  } |

（3）修改specificationController.js的save方法

|  |
| --- |
| //保存  $scope.save=**function**(){  **var** serviceObject;//服务层对象  **if**($scope.entity.specification.id!=**null**){//如果有ID  serviceObject=specificationService.update( $scope.entity ); //修改  }**else**{  serviceObject=specificationService.add( $scope.entity );//增加  }  serviceObject.success(  **function**(response){  **if**(response.success){  //重新查询  $scope.reloadList();//重新加载  }**else**{  alert(response.message);  }  }  );  } |

## 3.5删除规格 (同学完成了)

实现思路：我们要删除规格的同时，还要记得将关联的规格选项删除掉。



### **3.5.1 后端代码**

修改sellergoods的SpecificationServiceImpl.java

|  |
| --- |
| /\*\*  \* 批量删除  \*/  @Override  //删除规格表 删除规格属性表  **public** **void** delete(Long[] ids){  **for** (Long id : ids) {  specificationDao.deleteByPrimaryKey(id);  //删除规格属性表  SpecificationOptionQuery query = **new** SpecificationOptionQuery();  query.createCriteria().andSpecIdEqualTo(id);  specificationOptionDao.deleteByExample(query);  }  } |

### **3.5.2 前端代码**

修改pinyougou-manager-web的specification.html

列表的复选框

|  |
| --- |
| <input type=*"checkbox"* **ng-click=*"updateSelection($event,entity.id)"***> |

删除按钮

|  |
| --- |
| <button type=*"button"* class=*"btn btn-default"* title=*"删除"* **ng-click=*"dele()"*** ><i class=*"fa fa-trash-o"*></i> 删除</button> |

# 4.模板管理

## 4.1 需求及表结构分析

### **4.1.1 需求分析**

首先我们需要理解模板的作用。模板主要有两个：

1是用于关联品牌与规格

2定义扩充属性

### **4.1.2 表结构分析**

**tb\_type\_template 模板表**

|  |  |  |  |
| --- | --- | --- | --- |
| 字段 | 类型 | 长度 | 含义 |
| Id | Bigint |  | 主键 |
| name | Varchar | 80 | 模板名称 |
| Spec\_ids | Varchar | 1000 | 关联规格（json格式） |
| brand\_ids | Varchar | 1000 | 关联品牌（json格式） |
| custom\_attribute\_items | Varchar | 2000 | 扩展属性 |

## 4.2 模板列表

### **4.2.1 引入JS**

修改type\_template.html ，引入JS

|  |
| --- |
| <link rel=*"stylesheet"* href=*"../plugins/select2/select2.css"* />  <link rel=*"stylesheet"* href=*"../plugins/select2/select2-bootstrap.css"* />  <script src=*"../plugins/select2/select2.min.js"* type=*"text/javascript"*></script>  <script type=*"text/javascript"* src=*"../plugins/angularjs/angular.min.js"*></script>  <!-- 分页组件开始 -->  <script src=*"../plugins/angularjs/pagination.js"*></script>  <link rel=*"stylesheet"* href=*"../plugins/angularjs/pagination.css"*>  <!-- 分页组件结束 -->  <script type=*"text/javascript"* src=*"../js/base\_pagination.js"*></script>  <script type=*"text/javascript"* src=*"../js/angular-select2.js"*> </script>  <script type=*"text/javascript"* src=*"../js/service/typeTemplateService.js"*></script>  <script type=*"text/javascript"* src=*"../js/service/brandService.js"*></script>  <script type=*"text/javascript"* src=*"../js/service/specificationService.js"*></script>  <script type=*"text/javascript"* src=*"../js/controller/baseController.js"*></script>  <script type=*"text/javascript"* src=*"../js/controller/typeTemplateController.js"*></script> |

### **4.2.2 放置分页组件**

|  |
| --- |
| <tm-pagination conf=*"paginationConf"*></tm-pagination> |

### **4.2.3 sellerGoods 的TypeTemplateServiceImpl**

|  |
| --- |
| /\*\*  \* 模板管理  \* **@author** lx  \*  \*/  @Service  **public** **class** TypeTemplateServiceImpl **implements** TypeTemplateService {  @Autowired  **private** TypeTemplateDao typeTemplateDao;  //查询  **public** PageResult search(TypeTemplate typeTemplate,Integer pageNum,Integer pageSize){  PageHelper.*startPage*(pageNum, pageSize);  Page<TypeTemplate> page = (Page<TypeTemplate>) typeTemplateDao.selectByExample(**null**);  **return** **new** PageResult(page.getTotal(), page.getResult());  }  } |

### TypeTemplateController

|  |
| --- |
| /\*\*  \* 模块管理  \* **@author** lx  \*  \*/  @RestController  **public** **class** TypeTemplateController {    @Reference  **private** TypeTemplateService typeTemplateService;  //根据条件查询模板分页对象  @RequestMapping(value = "/typeTemplate/search.do")  **public** PageResult search(@RequestBody TypeTemplate typeTemplate,Integer page,Integer rows){  **return** typeTemplateService.search(typeTemplate, page, rows);    }  } |

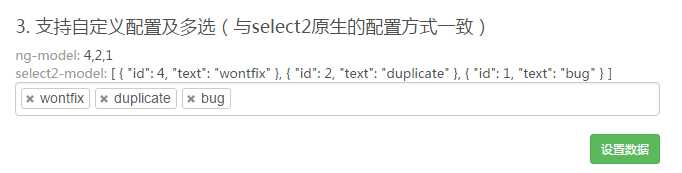
## 4.3 品牌下拉列表

在弹出窗口中有个品牌下拉列表，要求品牌是可以选择多个，这与我们之前的单选的下拉列表是不同的。我们要想实现这个功能，需要使用**select2** 组件来完成。



### **4.2.1 认识select2**

我们来看例子：我们需要的就是这样可以多选的下拉框



### **4.2.2 显示品牌下拉列表（静态）**

（1）修改 type\_template.html 引入JS, 上面已经一次性引入过了.这里不用再次操作.

1. 修改typeTemplateController.js ，定义品牌列表数据

|  |
| --- |
| $scope.brandList={data:[{id:1,text:'联想'},{id:2,text:'华为'},{id:3,text:'小米'}]};//品牌列表 |

1. 在type\_template.html 用select2组件实现多选下拉框

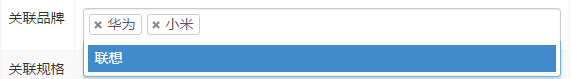
|  |
| --- |
| <input select2 select2-model=*"entity.brandIds"* config=*"brandList"* multiple placeholder=*"选择品牌（可多选）"* class=*"form-control"* type=*"text"*/> |

multiple 表示可多选

Config用于配置数据来源

select2-model用于指定用户选择后提交的变量

最终实现效果如下：



### **4.2.3 后端数据支撑**

我们现在让这个下拉列表的数据从数据库中提取，修改后端代码

（1）pinyougou-dao 工程 ，在BrandMapper.xml中添加SQL语句配置

|  |
| --- |
| <select id=*"selectOptionList"* resultType=*"java.util.Map"* >  select id,name as text from tb\_brand  </select> |

（2）在pinyougou-dao 的BrandMapper中添加方法定义

|  |
| --- |
| List<Map> selectOptionList(); |

（3）修改pinyougou-sellergoods-interface 的BrandService.java，增加方法定义

|  |
| --- |
| /\*\*  \* 品牌下拉框数据  \*/  List<Map> selectOptionList(); |

（4）修改sellergoods的BrandServiceImpl.java，增加方法

|  |
| --- |
| /\*\*  \* 列表数据  \*/  **public** List<Map> selectOptionList() {  **return** brandDao.selectOptionList();  } |

（5）修改manager的BrandController.java

|  |
| --- |
| @RequestMapping("/selectOptionList")  **public** List<Map> selectOptionList(){  **return** brandService.selectOptionList();  } |

1. 修改manager的brandService.js

|  |
| --- |
| //下拉列表数据  **this**.selectOptionList=**function**(){  **return** $http.get('../brand/selectOptionList.do');  } |

1. 修改manager的typeTemplateController.js

因为我们在模板控制层中需要使用品牌服务层的方法，所以需要添加**依赖注入**

|  |
| --- |
| //控制层  app.controller('typeTemplateController' ,**function**($scope,$controller ,typeTemplateService ,brandService){ |

使用品牌服务方法实现查询，结果赋给变量

|  |  |
| --- | --- |
| $scope.brandList={data:[]};//品牌列表  //读取品牌列表  $scope.findBrandList=**function**(){  brandService.selectOptionList().success(  **function**(response){  $scope.brandList={data:response};  }  );  } |  |

type\_template.html加入下面代码

|  |
| --- |
| <body class=*"hold-transition skin-red sidebar-mini"*  ng-app=*"pinyougou"* ng-controller=*"typeTemplateController"* ng-init=*"findBrandList()"*> |

## 4.4 规格下拉列表



我们现在让这个下拉列表的数据从数据库中提取，修改后端代码

（1）pinyougou-dao 工程 ，在SpecificationDao.xmll中添加SQL语句配置

|  |
| --- |
| <select id=*"selectOptionList"* resultType=*"java.util.Map"* >  select id,spec\_name as text from tb\_specification  </select> |

（2）在pinyougou-dao 的SpecificationDao.java中添加方法定义

|  |
| --- |
| List<Map> selectOptionList(); |

（3）修改sellergoods 的SpecificationService.java，增加方法定义

|  |
| --- |
| List<Map> selectOptionList(); |

（4）修改sellergoods的SpecificationServiceImpl.java，增加方法

|  |
| --- |
| **public** List<Map> selectOptionList() {  **return** specificationDao.selectOptionList();  } |

（5）修改pinyougou-manager-web的SpecificationController.java

|  |
| --- |
| @RequestMapping("/selectOptionList")  **public** List<Map> selectOptionList() **throws** Exception {  **return** specificationService.selectOptionList();  } |

1. 修改manager的specificationService.js

|  |
| --- |
| //下拉列表  **this**.selectOptionList=**function**(){  **return** $http.get('../specification/selectOptionList.do');  } |

1. 修改manager的typeTemplateController.js

因为我们在模板控制层中需要使用规格服务层的方法，所以需要添加**依赖注入**

|  |
| --- |
| //控制层  app.controller('typeTemplateController' ,**function**($scope,$controller ,typeTemplateService,brandService,specificationService){ |

使用规格服务方法实现查询，结果赋给变量

|  |
| --- |
| $scope.specList={data:[]};//规格列表  //读取规格列表  $scope.findSpecList=**function**(){  specificationService.selectOptionList().success(  **function**(response){  $scope.specList={data:response};  }  );  } |

type\_template.html页面加入下面代码

|  |
| --- |
| <body class=*"hold-transition skin-red sidebar-mini"*  ng-app=*"pinyougou"* ng-controller=*"typeTemplateController"* ng-init=*"findBrandList();findSpecList()"*>  <tr>  <td>关联规格</td>  <td>  <input select2 select2-model=*"entity.specIds"* config=*"specList"* multiple placeholder=*"支持多选哦"* class=*"form-control"* type=*"text"*/>  </td>  </tr> |

## 4.5 扩展属性

### **4.5.1 增加行**

在typeTemplateController.js中新增代码

|  |
| --- |
| //新增扩展属性行  $scope.addTableRow=**function**(){  $scope.entity.customAttributeItems.push({});  } |

在type\_template.html中的“新建”按钮，执行实体的初始化操作

|  |
| --- |
| <button type=*"button"* class=*"btn btn-default"* title=*"新建"* data-toggle=*"modal"* data-target=*"#editModal"* **ng-click=*"entity={customAttributeItems:[]}"***><i class=*"fa fa-file-o"*></i> 新建</button> |

修改“新增扩展属性按钮”

|  |
| --- |
| <button type=*"button"* class=*"btn btn-default"* title=*"新增扩展属性"* **ng-click=*"addTableRow()"***><i class=*"fa fa-file-o"*></i> 新增扩展属性</button> |

循环表格

|  |
| --- |
| <tr ng-repeat=*"pojo in entity.customAttributeItems"*>  <td><input type=*"checkbox"* class=*"icheckbox\_square-blue"* ></td>  <td><input class=*"form-control"* placeholder=*"属性名称"* ng-model=*"pojo.text"*></td>  <td><button type=*"button"* class=*"btn btn-default"* title=*"删除"* ng-click=*"deleTableRow($index)"*><i class=*"fa fa-trash-o"*></i> 删除</button></td>  </tr> |

### **4.5.2 删除行**

实现思路：在每一行将索引值传递给集合，在集合中删除。

修改typeTemplateController.js新增以下代码

|  |
| --- |
| //删除扩展属性行  $scope.deleTableRow=**function**(index){  $scope.entity.customAttributeItems.splice(index,1);//删除  } |

修改每行的删除按钮

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

$index 用于获取ng-repeat指令循环中的索引。

## 4.6 新增模板

修改type\_template.html ，绑定文本框

|  |
| --- |
| <tr>  <td>模板类型</td>  <td><input ng-model=*"entity.name"* class=*"form-control"* placeholder=*"模板类型"*> </td>  </tr> |

保存按钮

|  |
| --- |
| <button class=*"btn btn-success"* data-dismiss=*"modal"* aria-hidden=*"true"* ng-click=*"save()"*>保存</button> |

## 4.7 修改模板

修改typeTemplateController.js的findOne方法

|  |
| --- |
| //查询实体  $scope.findOne=**function**(id){  typeTemplateService.findOne(id).success(  **function**(response){  $scope.entity= response;  $scope.entity.brandIds= JSON.parse($scope.entity.brandIds);//转换品牌列表  $scope.entity.specIds= JSON.parse($scope.entity.specIds);//转换规格列表  $scope.entity.customAttributeItems= JSON.parse($scope.entity.customAttributeItems);//转换扩展属性  }  );  } |

从数据库中查询出来的是字符串，我们必须将其转换为json对象才能实现信息的回显。

## 4.8 删除模板

修改type\_template.html

表格中的复选框

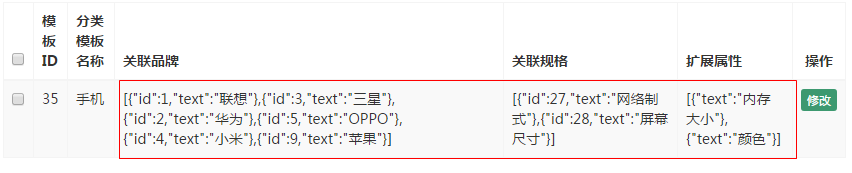
|  |
| --- |
| <input type=*"checkbox"* ng-click=*"updateSelection($event,entity.id)"*> |

删除按钮

|  |
| --- |
| <button type=*"button"* class=*"btn btn-default"* title=*"删除"* **ng-click=*"dele()"***>  <i class=*"fa fa-trash-o"*></i> 删除</button> |

## 4.9 优化模板列表的显示

我们现在完成的列表中都是以JSON格式显示的，不利于用户的查询。



我们需要将信息以更友好的方式展现出来,如下图形式



我们需要将一个json字符串中某个属性的值提取出来，用逗号拼接成一个新的字符串。这样的功能比较常用，所以我们将方法写到baseController.js

|  |
| --- |
| //提取json字符串数据中某个属性，返回拼接字符串 逗号分隔  $scope.jsonToString=**function**(jsonString,key){  **var** json=JSON.parse(jsonString);//将json字符串转换为json对象  **var** value="";  **for**(**var** i=0;i<json.length;i++){  **if**(i>0){  value+=","  }  value+=json[i][key];  }  **return** value;  } |

页面上使用该函数进行转换

|  |
| --- |
| <tr ng-repeat=*"entity in list"*>  <td><input type=*"checkbox"* ng-click=*"updateSelection($event,entity.id)"*></td>  <td>{{entity.id}}</td>  <td>{{entity.name}}</td>  <td>{{jsonToString(entity.brandIds,'text')}}</td>  <td>{{jsonToString(entity.specIds,'text')}}</td> <td>{{jsonToString(entity.customAttributeItems,'text')}}</td>  <td class=*"text-center"*>  <button type=*"button"* class=*"btn bg-olive btn-xs"* data-toggle=*"modal"* data-target=*"#editModal"* ng-click=*"findOne(entity.id)"*>修改</button>  </td>  </tr> |