### EasyUI异步加载树要求

#### 返回json对象格式要求

* id: 节点id,它是重要的来加载远程数据
* text: 节点文本来显示
* state: 节点状态,“open”或“closed”,默认是“open”。当设置为“closed”,节点有子节点,并将负载从远程站点

需要POJO对象增加两个属性，以满足返回的json格式要求

// 扩展get方法，满足EasyUI的tree格式

public String getText() {

return getName();

}

public String getState() {

return getIsParent() ? "closed" : "open";

}

#### EasyUI异步加载树工作原理

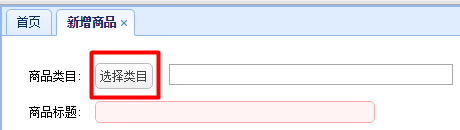
用户创建一个空的树，然后指定一个服务器端动态返回JSON数据用来填充树与异步和需求。子节点加载依赖父节点状态。当扩展一个封闭的节点，如果节点没有子节点加载，它将发送节点id的值作为http请求参数命名为“id”到服务器上面定义通过URL检索的子节点。

### 商品分类树加载执行过程

用户点击EasyUI的selectItemCat按钮，调用页面加载时的jQuery的KindEditorUtil.init()方法，KindEditorUtil为在common.js中定义的js对象。在init方法中调用this.initItemCat(data)。Each遍历找到selectItemCat元素，给其绑定onclick事件，onclick事件中创建window窗口，在窗口的ul上加载EasyUITree。异步post提交，通过url（/item/cat/list）指定调用路径，通过注解找到ItemCatController，执行它的queryList方法，controller中调用service层，执行ItemCatService的queryListById方法，service中调用mybatis的mapper接口，调用 queryListById方法，其执行ItemCatMapper.xml文件中配置的queryListById。执行SQL语句。查询后返回ItemCat对象集合，并由springmvc的注解@ResponseBody将java对象转换为json串，最终返回给js，EasyUI解析并显示树形结构。

### 开发步骤

EasyUI.tree树为异步加载，点开树枝，才显示下级节点。将传递参数，父节点的id，默认第一次页面传值为0。





第一步：在item-add.jsp页面中调用js事件。

<tr>

<td>商品类目:</td>

<td>

<a href="javascript:void(0)" class="easyui-linkbutton selectItemCat">选择类目</a>

<input type="hidden" name="cid" style="width: 280px;"></input>

</td>

</tr>

在a标签上的class属性中，绑定了easyui的按钮，设定class名称为selectItemCat，在js中将利用这个class来绑定onclick事件。隐藏域用来返回用户选择树节点的值。

第二步：调用common.js中的initItemCat

……

initItemCat : function(data){

$(".selectItemCat").each(function(i,e){//i= index 下标，e：element：元素

var \_ele = $(e);

if(data && data.cid){

\_ele.after("<span style='margin-left:10px;'>"+data.cid+"</span>");

}else{

\_ele.after("<span style='margin-left:10px;'></span>");

}

\_ele.unbind('click').click(function(){

$("<div>").css({padding:"5px"}).html("<ul>")

.window({

width:'500',

height:"450",

modal:true,

closed:true,

iconCls:'icon-save',

title:'选择分类',

onOpen : function(){ //当窗口打开后执行

var \_win = this; //this就指向当前的window

$("ul",\_win).tree({

url:'/item/cat/list',

animate:true,

onClick : function(node){

//只有当是叶子节点时，才进行处理

if($(this).tree("isLeaf",node.target)){

// 填写到cid中

//\_ele.parent，在当前元素的父对象，目的缩小查询范围 \_ele.parent().find("[name=cid]").val(node.id);

//find(“[name=cid]”)通过jQuery的选择器，通过一个属性进行选择name=cid的

//元素后面，加上选择节点的文字。

\_ele.next().text(node.text).attr("cid",node.id);

$(\_win).window('close');

if(data && data.fun){

data.fun.call(this,node);

}

}

}

});

},

onClose : function(){

$(this).window("destroy");

}

}).window('open');

});

});

},

……

第三步：pojo

创建ItemCat pojo继承BasePojo；增加getText方法，为easy准备。

package com.jt.manage.pojo;

import javax.persistence.Column;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

import javax.persistence.Table;

@Table(name = "tb\_item\_cat")

public class ItemCat extends BasePojo{

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

// 主键自增长策略

private Long id;

// 父类目ID=0时，代表的是一级的类目

@Column(name = "parent\_id")

private Long parentId;

private String name;

// 状态。可选值:1(正常),2(删除)

private Integer status;

// 排列序号，表示同级类目的展现次序，如数值相等则按名称次序排列。取值范围:大于零的整数

@Column(name = "sort\_order")

private Integer sortOrder;

// 该类目是否为父类目，1为true，0为false

@Column(name = "is\_parent")

private Boolean isParent;

public Long getId() {

return id;

}

public void setId(Long id) {

this.id = id;

}

public Long getParentId() {

return parentId;

}

public void setParentId(Long parentId) {

this.parentId = parentId;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public Integer getStatus() {

return status;

}

public void setStatus(Integer status) {

this.status = status;

}

public Integer getSortOrder() {

return sortOrder;

}

public void setSortOrder(Integer sortOrder) {

this.sortOrder = sortOrder;

}

public Boolean getIsParent() {

return isParent;

}

public void setIsParent(Boolean isParent) {

this.isParent = isParent;

}

// 扩展get方法，满足EasyUI的tree格式

public String getText() {

return getName();

}

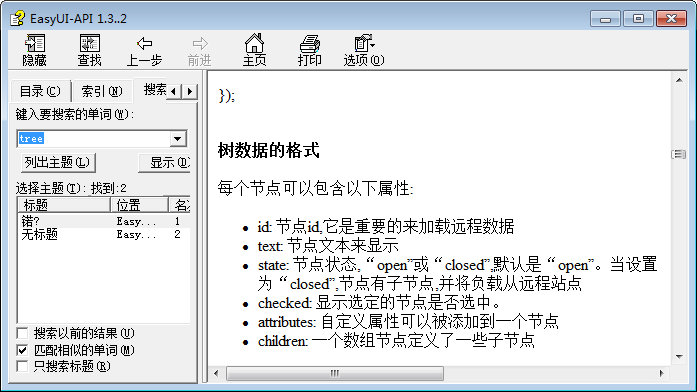
public String getState() {

return getIsParent() ? "closed" : "open";

}

}

D:\0202\project\24-ds\01-课前资料\21-EasyUI\EasyUI-API+1.3.2.chm



第四步：controller层

package com.jt.manage.controller;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestParam;

import org.springframework.web.bind.annotation.ResponseBody;

import com.jt.manage.pojo.ItemCat;

import com.jt.manage.service.ItemCatService;

/\*\*

\* 商品类目相关的业务逻辑处理

\*

\*/

@RequestMapping("/item/cat")

@Controller

public class ItemCatController {

@Autowired

private ItemCatService itemCatService;

@RequestMapping("list")

@ResponseBody

public List<ItemCat> queryList(@RequestParam(value="id", defaultValue="0") Long id){

return this.itemCatService.queryListById(id);

}

}

第五步：service层

package com.jt.manage.service;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

import com.jt.manage.mapper.ItemCatMapper;

import com.jt.manage.pojo.ItemCat;

@Service

public class ItemCatService extends BaseService<ItemCat> {

@Autowired

private ItemCatMapper itemCatMapper;

public List<ItemCat> queryListById(Long id) {

return this.itemCatMapper.queryListById(id);

}

}

第六步：interface接口

package com.jt.manage.mapper;

import java.util.List;

import org.apache.ibatis.annotations.Param;

import com.jt.manage.mapper.base.mapper.SysMapper;

import com.jt.manage.pojo.ItemCat;

public interface ItemCatMapper extends SysMapper<ItemCat>{

/\*\*

\* 根据ID查询商品分类数据，parentId = id(参数)

\*

\* @param id

\* @return

\*/

List<ItemCat> queryListById(@Param("id") Long id);

}

第七步：mapper映射文件

<?xml version="1.0" encoding="UTF-8" ?>

<!DOCTYPE mapper

PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"

"http://mybatis.org/dtd/mybatis-3-mapper.dtd">

<mapper namespace="com.jt.manage.mapper.ItemCatMapper">

<select id="queryItemCat" parameterType="long" resultType="ItemCat">

SELECT \* FROM tb\_item\_cat WHERE STATUS=1 AND parent\_id=#{id}

ORDER BY parent\_id,sort\_order

</select>

</mapper>