## 部门管理界面的开发

### 1.1引入ACE模板

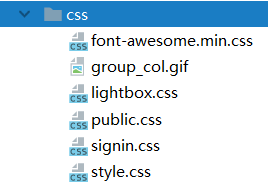


assets包含了ace使用的js css及图片信息。

### 1.2引入Bootstrap



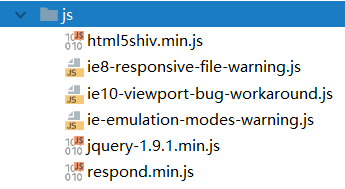
### 1.3引入常用的css



font-weasome是一套绝佳的图标字体库和css框架。提供可缩放的矢量图标，可以使用css所提供的所有特性对它们进行更改，包括：大小、颜色、阴影或者其它任何支持的效果。

lightbox.css灯箱效果

### 1.4引入js



html5shiv.min.js是一个针对IE浏览器的HTML5JavaScript补丁，目的是让IE识别并支持HTML5元素。

ie8-responsive-file-warning.js兼容ie8

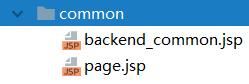
ie10-viewport-bug-workaround.js兼容ie10

ie-emulation-modes-warning.js通用兼容

jquery自用

rspond.min.js是一个快速、轻量的polyfill，用于为IE6-8以及其它不支持css3media Queries的浏览器提供媒体查询的min-width和max-width特性，实现响应式网页设计（Responsive Web Design）。

### 1.5创建通用页面



backend\_common.jsp中引入页面要用到的所有css和js文件。

<%@ **page contentType**="**text/html;charset=UTF-8**" **language**="**java**" %>  
  
<**meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1"**/>  
<**meta http-equiv="pragma" content="no-cache"**/>  
<**meta charset="utf-8"**/>  
<**title**>管理员控制台</**title**>  
<**meta name="description" content="overview &amp; stats"**/>  
<**meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0"**/>  
<**link rel="shortcut icon" href="/favicon.ico" type="image/x-icon"**>  
  
*<!-- bootstrap & fontawesome -->*<**link rel="stylesheet" href="/assets/css/bootstrap.min.css"**/>  
<**link rel="stylesheet" href="http://netdna.bootstrapcdn.com/font-awesome/4.0.3/css/font-awesome.min.css"**/>  
  
*<!-- page specific plugin styles -->  
<!-- text fonts -->*<**link rel="stylesheet" href="/assets/css/google-fonts.css"**/>  
  
*<!-- ace styles -->*<**link rel="stylesheet" href="/assets/css/ace.min.css"**/>  
  
*<!--[if lte IE 9]>  
<link rel="stylesheet" href="/assets/css/ace-part2.min.css"/>  
  
<![endif]-->*<**link rel="stylesheet" href="/assets/css/ace-skins.min.css"**/>  
<**link rel="stylesheet" href="/assets/css/ace-rtl.min.css"**/>  
<**link rel="stylesheet" href="/assets/css/jquery.gritter.css"**/>  
<**link rel="stylesheet" href="/assets/css/jquery-ui.min.css"**/>  
<**link rel="stylesheet" href="/assets/css/chosen.css"**/>  
  
*<!--[if lte IE 9]>  
<link rel="stylesheet" href="/assets/css/ace-ie.min.css"/>  
  
<![endif]-->  
<!-- inline styles related to this page -->  
<!-- ace settings handler -->*<**script src="/assets/js/ace-extra.min.js"**></**script**>  
  
*<!-- HTML5 shim and Respond.js IE8 support of HTML5 elements and media queries -->  
<!--[if lte IE 8]>  
<script src="/js/html5shiv.min.js"></script>  
<script src="/js/respond.min.js"></script>  
  
<![endif]-->*<**script src="/js/jquery-1.9.1.min.js"**></**script**>  
<**script src="/assets/js/jquery-ui.min.js"**></**script**>  
<**script src="/assets/js/jquery.gritter.min.js"**></**script**>  
<**script src="/assets/js/chosen.jquery.min.js"**></**script**>  
<**script src="http://cdn.bootcss.com/mustache.js/2.2.1/mustache.js"**></**script**>  
<**script src="/bootstrap3.3.5/js/bootstrap.min.js"**></**script**>  
  
<**script type="text/javascript"**>  
 *// 展示提示信息* **function** *showMessage*(title, msg, isSuccess) {  
 **if** (!isSuccess) {  
 msg = msg || **''**;  
 } **else** {  
 msg = msg || **'操作成功'** }  
 **$**.**gritter**.add({  
 **title**: title,  
 **text**: msg != **''** ? msg : **"服务器处理异常, 建议刷新页面来保证数据是最新的"**,  
 **time**: **''**,  
 **class\_name**: (isSuccess ? **'gritter-success'** : **'gritter-warning'**) + (!**$**(**'#gritter-light'**).get(0).**checked** ? **' gritter-light'** : **''**)  
 });  
 }  
</**script**>

其中添加一个展示提示信息的方法showMessage(title, msg, isSuccess)

Gritter是一个小型的jQuery消息通知插件。

### 1.6创建dept.jsp

在<head>标签里引入backend\_common.jsp页面；

### 1.7在spring-servlet.xml里添加配置

页面上的js，css文件等相关元素需要进行配置

*<!--如果在配置前端控制器时拦截了所有的请求，不做特殊处理就会导致部分静态资源无法使用。  
如果是这种情况就可以使用下面的配置来访问静态资源文件-->*<**mvc:resources location="/js/" mapping="/js/\*\*"**/>  
<**mvc:resources location="/css/" mapping="/css/\*\*"**/>  
<**mvc:resources location="/bootstrap3.3.5/" mapping="/bootstrap3.3.5/\*\*"**/>  
<**mvc:resources location="/assets/" mapping="/assets/\*\*"**/>  
<**mvc:resources location="/ztree/" mapping="/ztree/\*\*"**/>

### 1.8部门树的展示

<**div class="col-sm-3"**>  
 <**div class="table-header"**>  
 部门列表**&nbsp;&nbsp;** <**a class="green" href="#"**>  
 *<%--新增部门按钮--%>* <**i class="ace-icon fa fa-plus-circle orange bigger-130 dept-add"**></**i**>  
 </**a**>  
 </**div**>  
 <**div id="deptList"**>  
 *<%--展示部门列表--%>* </**div**>  
</**div**>

定义部门列表的mustache模板：

*<%--mustache模板--%>*<**script id="deptListTemplate" type="x-tmpl-mustache"**>  
<**ol class="dd-list"**>  
 *<%--模板中使用循环，循环deptList的元素--%>* {{#deptList}}  
 <li class="dd-item dd2-item dept-name" id="dept\_{{id}}" href="javascript:void(0)" data-id="{{id}}">  
 *<%--部门名称--%>* <div class="dd2-content" style="cursor:pointer;">  
 {{name}}  
 *<%--编辑、删除按钮--%>* <span style="float:right;">  
 <a class="green dept-edit" href="#" data-id="{{id}}" >  
 <i class="ace-icon fa fa-pencil bigger-100"></i>  
 </a>  
 &nbsp;  
 <a class="red dept-delete" href="#" data-id="{{id}}" data-name="{{name}}">  
 <i class="ace-icon fa fa-trash-o bigger-100"></i>  
 </a>  
 </span>  
 </div>  
 </li>  
 {{/deptList}}  
</ol>  
</**script**>

使用jQuery脚本来处理：

**$**(**function** () {  
 **var** deptList; *// 存储树形部门列表* **var** deptMap = {}; *// 存储map格式的部门信息* **var** userMap = {}; *// 存储map格式的用户信息* **var** optionStr = **""**; *//下拉框的选项* **var** lastClickDeptId = -1;  
  
 **var** deptListTemplate = **$**(**'#deptListTemplate'**).html();  
 Mustache.parse(deptListTemplate);  
 **var** userListTemplate = **$**(**'#userListTemplate'**).html();  
 Mustache.parse(userListTemplate);  
  
 *loadDeptTree*(); *//初始化并显示部门树  
  
 //渲染第一层部门树* **function** *loadDeptTree*() {  
 **$**.ajax({  
 **url**: **"/sys/dept/tree.json"**,  
 success: **function** (result) {  
 **if** (result.ret) {  
 deptList = result.**data**; *//deptList存到全局变量* **var** rendered = Mustache.render(deptListTemplate, {**deptList**: result.**data**}); *//把数据deptList（来源：后台返回的json数据）渲染到deptListTemplate模板用作循环* **$**(**"#deptList"**).html(rendered); *//把渲染结果插入空的<div id="deptList"></div>下显示  
 recursiveRenderDept*(result.**data**);  
 *bindDeptClick*(); *//绑定修改、删除、加载当前部门的用户列表点击事件* } **else** {  
 *showMessage*(**"加载部门列表"**, result.**msg**, **false**);  
 }  
 }  
 })  
 }  
  
 *// 递归渲染部门树* **function** *recursiveRenderDept*(deptList) {  
 **if** (deptList && deptList.**length** > 0) {  
 **$**(deptList).each(**function** (i, dept) {  
 deptMap[dept.**id**] = dept; *//所有的部门信息存到全局变量  
 //如果存在子部门递归处理* **if** (dept.**deptList**.**length** > 0) {  
 **var** rendered = Mustache.render(deptListTemplate, {**deptList**: dept.**deptList**});  
 **$**(**"#dept\_"** + dept.**id**).append(rendered);  
 *recursiveRenderDept*(dept.**deptList**);  
 }  
 })  
 }  
 }

### 1.9新增部门

<**div class="table-header"**>  
 部门列表**&nbsp;&nbsp;** <**a class="green" href="#"**>  
 *<%--新增部门按钮--%>* <**i class="ace-icon fa fa-plus-circle orange bigger-130 dept-add"**></**i**>  
 </**a**>  
</**div**>

*<%--新增部门对话框--%>*<**div id="dialog-dept-form" style="display**: **none**;**"**>  
 <**form id="deptForm"**>  
 <**table class="table table-striped table-bordered table-hover dataTable no-footer" role="grid"**>  
 <**tr**>  
 <**td style="width**: 80**px**;**"**><**label for="parentId"**>上级部门</**label**></**td**>  
 <**td**>  
 <**select id="parentId" name="parentId" data-placeholder="选择部门" style="width**: 200**px**;**"**></**select**>  
 <**input type="hidden" name="id" id="deptId"**/>  
 </**td**>  
 </**tr**>  
 <**tr**>  
 <**td**><**label for="deptName"**>名称</**label**></**td**>  
 <**td**><**input type="text" name="name" id="deptName" value="" class="text ui-widget-content ui-corner-all"**>  
 </**td**>  
 </**tr**>  
 <**tr**>  
 <**td**><**label for="deptSeq"**>顺序</**label**></**td**>  
 <**td**><**input type="text" name="seq" id="deptSeq" value="1" class="text ui-widget-content ui-corner-all"**>  
 </**td**>  
 </**tr**>  
 <**tr**>  
 <**td**><**label for="deptRemark"**>备注</**label**></**td**>  
 <**td**><**textarea name="remark" id="deptRemark" class="text ui-widget-content ui-corner-all" rows="3"  
 cols="25"**></**textarea**></**td**>  
 </**tr**>  
 </**table**>  
 </**form**>  
</**div**>

*//点击增加按钮弹出增加部门模态对话框***$**(**".dept-add"**).click(**function** () {  
 **$**(**"#dialog-dept-form"**).**dialog**({  
 **model**: **true**, *//是否为模态对话框* **title**: **"新增部门"**,  
 *//对话框打开时数据等初始化* open: **function** (event, ui) {  
 **$**(**".ui-dialog-titlebar-close"**, **$**(**this**).parent()).hide(); *//取消所有默认功能（关闭）按钮，原因是关闭时可以处理相关操作并不是单纯的关闭* optionStr = **"<option value=\"0\">-</option>"**;  
 *recursiveRenderDeptSelect*(deptList, 1);  
 **$**(**"#deptForm"**)[0].reset(); *//重置表格，清空表格数据* **$**(**"#parentId"**).html(optionStr); *//渲染到<select id="parentId"></select>中* },  
 *//属性名将会被作为按钮的提示文字，属性值为一个函数，即按钮的处理函数* **buttons**: {  
 **"添加"**: **function** (e) {  
 e.preventDefault(); *//拦截默认的点击事件  
 //true为新增，false为修改。操作成功关闭模态框  
 updateDept*(**true**, **function** (data) {  
 **$**(**"#dialog-dept-form"**).**dialog**(**"close"**);  
 }, **function** (data) {  
 *showMessage*(**"新增部门"**, data.**msg**, **false**);  
 })  
 },  
 **"取消"**: **function** () {  
 **$**(**"#dialog-dept-form"**).**dialog**(**"close"**);  
 }  
 }  
 });  
});  
  
*//下拉框初始化部门列表函数***function** *recursiveRenderDeptSelect*(deptList, level) {  
 level = level | 0; *//level没有则为置为0* **if** (deptList && deptList.**length** > 0) {  
 **$**(deptList).each(**function** (i, dept) {  
 deptMap[dept.**id**] = dept;  
 **var** blank = **""**;  
 *//如果level不是第一层* **if** (level > 1) {  
 **for** (**var** j = 3; j <= level; j++) {  
 blank += **".."**;  
 }  
 blank += **"∟"**;  
 }  
 *//渲染数据* optionStr += Mustache.render(**"<option value='{{id}}'>{{name}}</option>"**, {  
 **id**: dept.**id**,  
 **name**: blank + dept.**name** });  
 *//递归渲染* **if** (dept.**deptList** && dept.**deptList**.**length** > 0) {  
 *recursiveRenderDeptSelect*(dept.**deptList**, level + 1);  
 }  
 });  
 }  
}

*//部门（保存、更新）函数***function** *updateDept*(isCreate, successCallback, failCallback) {  
 **$**.ajax({  
 **url**: isCreate ? **"/sys/dept/save.json"** : **"/sys/dept/update.json"**,  
 **data**: **$**(**"#deptForm"**).serializeArray(),  
 **type**: **'POST'**,  
 success: **function** (result) {  
 **if** (result.ret) {  
 *loadDeptTree*();  
 **if** (successCallback) {  
 successCallback(result);  
 }  
 } **else** {  
 **if** (failCallback) {  
 failCallback(result);  
 }  
 }  
 }  
 })  
}

### 1.10编辑部门

*<%--mustache模板--%>*<**script id="deptListTemplate" type="x-tmpl-mustache"**>  
<**ol class="dd-list"**>  
 *<%--模板中使用循环，循环deptList的元素--%>* {{#deptList}}  
 <li class="dd-item dd2-item dept-name" id="dept\_{{id}}" href="javascript:void(0)" data-id="{{id}}">  
 *<%--部门名称--%>* <div class="dd2-content" style="cursor:pointer;">  
 {{name}}  
 *<%--编辑、删除按钮--%>* <span style="float:right;">  
 <a class="green dept-edit" href="#" data-id="{{id}}" >  
 <i class="ace-icon fa fa-pencil bigger-100"></i>  
 </a>  
 &nbsp;  
 <a class="red dept-delete" href="#" data-id="{{id}}" data-name="{{name}}">  
 <i class="ace-icon fa fa-trash-o bigger-100"></i>  
 </a>  
 </span>  
 </div>  
 </li>  
 {{/deptList}}  
</ol>  
</**script**>

*//点击修改按钮弹出修改模态对话框* **$**(**".dept-edit"**).click(**function** (e) {  
 e.preventDefault(); *//拦截默认的点击时间内* e.stopPropagation(); *//阻止冒泡时间* **var** deptId = **$**(**this**).attr(**"data-id"**); *//取出当前元素的data-id的值* **$**(**"#dialog-dept-form"**).**dialog**({  
 **model**: **true**, *//是否为模态对话框* **title**: **"编辑部门"**,  
 *//对话框打开时数据等初始化* open: **function** (event, ui) {  
 **$**(**".ui-dialog-titlebar-close"**, **$**(**this**).parent()).hide(); *//取消所有默认的功能按钮，原因是关闭时可以处理相关操作并不是单纯的关闭* optionStr = **"<option value=\"0\">-</option>"**;  
 *recursiveRenderDeptSelect*(deptList, 1);  
 **$**(**"#deptForm"**)[0].reset(); *//重置表格，清空表格数据* **$**(**"#parentId"**).html(optionStr); *//渲染到<select id="parentId"></select>中* **$**(**"#deptId"**).val(deptId); *//获取当前部门的deptId信息填入输入框* **var** targetDept = deptMap[deptId]; *//根据deptId从存储所有部门信息的deptMap取出* **if** (targetDept) {  
 **$**(**"#parentId"**).val(targetDept.**parentId**);  
 **$**(**"#deptName"**).val(targetDept.**name**);  
 **$**(**"#deptSeq"**).val(targetDept.**seq**);  
 **$**(**"#deptRemark"**).val(targetDept.**remark**);  
 }  
 },  
 *//buttons：属性名将会被作为按钮的提示文字，属性值为一个函数，即按钮的处理函数* **buttons**: {  
 **"更新"**: **function** (e) {  
 e.preventDefault();  
 *updateDept*(**false**, **function** (data) {  
 **$**(**"#dialog-dept-form"**).**dialog**(**"close"**);  
 }, **function** (data) {  
 *showMessage*(**"更新部门"**, data.**msg**, **false**);  
 })  
 },  
 **"取消"**: **function** () {  
 **$**(**"#dialog-dept-form"**).**dialog**(**"close"**);  
 }  
 }  
 });  
 })  
}

### 1.11删除部门

*<%--mustache模板--%>*<**script id="deptListTemplate" type="x-tmpl-mustache"**>  
<**ol class="dd-list"**>  
 *<%--模板中使用循环，循环deptList的元素--%>* {{#deptList}}  
 <li class="dd-item dd2-item dept-name" id="dept\_{{id}}" href="javascript:void(0)" data-id="{{id}}">  
 *<%--部门名称--%>* <div class="dd2-content" style="cursor:pointer;">  
 {{name}}  
 *<%--编辑、删除按钮--%>* <span style="float:right;">  
 <a class="green dept-edit" href="#" data-id="{{id}}" >  
 <i class="ace-icon fa fa-pencil bigger-100"></i>  
 </a>  
 &nbsp;  
 <a class="red dept-delete" href="#" data-id="{{id}}" data-name="{{name}}">  
 <i class="ace-icon fa fa-trash-o bigger-100"></i>  
 </a>  
 </span>  
 </div>  
 </li>  
 {{/deptList}}  
</ol>  
</**script**>

*//点击删除按钮删除数据***$**(**".dept-delete"**).click(**function** (e) {  
 e.preventDefault(); *//拦截默认的点击事件* e.stopPropagation(); *//阻止冒泡时间* **var** deptId = **$**(**this**).attr(**"data-id"**); *//取出当前元素的data-id的值* **var** deptName = **$**(**this**).attr(**"data-name"**); *//取出当前元素的data-name的值* **if** (*confirm*(**"确定要删除部门["** + deptName + **"]吗?"**)) {  
 **$**.ajax({  
 **url**: **"/sys/dept/delete.json"**,  
 **data**: {  
 **id**: deptId  
 },  
 success: **function** (result) {  
 **if** (result.ret) {  
 *showMessage*(**"删除部门["** + deptName + **"]"**, **"操作成功"**, **true**);  
 *loadDeptTree*();  
 } **else** {  
 *showMessage*(**"删除部门["** + deptName + **"]"**, result.**msg**, **false**);  
 }  
 }  
 });  
 }  
});

### 1.12加载部门信息

*<%--mustache模板--%>*<**script id="deptListTemplate" type="x-tmpl-mustache"**>  
<**ol class="dd-list"**>  
 *<%--模板中使用循环，循环deptList的元素--%>* {{#deptList}}  
 <li class="dd-item dd2-item dept-name" id="dept\_{{id}}" href="javascript:void(0)" data-id="{{id}}">  
 *<%--部门名称--%>* <div class="dd2-content" style="cursor:pointer;">  
 {{name}}  
 *<%--编辑、删除按钮--%>* <span style="float:right;">  
 <a class="green dept-edit" href="#" data-id="{{id}}" >  
 <i class="ace-icon fa fa-pencil bigger-100"></i>  
 </a>  
 &nbsp;  
 <a class="red dept-delete" href="#" data-id="{{id}}" data-name="{{name}}">  
 <i class="ace-icon fa fa-trash-o bigger-100"></i>  
 </a>  
 </span>  
 </div>  
 </li>  
 {{/deptList}}  
</ol>  
</**script**>

### 1.13加载当前部门的用户列表

*//加载当前部门的用户列表***$**(**".dept-name"**).click(**function** (e) {  
 e.preventDefault(); *//拦截默认的点击事件* e.stopPropagation(); *//阻止冒泡事件* **var** deptId = **$**(**this**).attr(**"data-id"**); *//取出当前元素的data-id的值  
 handleDepSelected*(deptId);  
});

*//根据当前deptId加载用户部门列表函数***function** *handleDepSelected*(deptId) {  
 **if** (lastClickDeptId != -1) {  
 **var** lastDept = **$**(**"#dept\_"** + lastClickDeptId + **" .dd2-content:first"**); *//获取上次点击的部门的<li>下显示部门名称的<div>节点* lastDept.removeClass(**"btn-yellow"**); *//去除高亮样式* lastDept.removeClass(**"no-hover"**);  
 }  
 **var** currentDept = **$**(**"#dept\_"** + deptId + **" .dd2-content:first"**); *//获取当前点击的部门的<li>下显示部门名称的<div>节点* currentDept.addClass(**"btn-yellow"**);  
 currentDept.addClass(**"no-hover"**);  
 lastClickDeptId = deptId; *//把当前点击的deptId设置为上次点击的deptId  
 loadUserList*(deptId); *//加载用户部门用户列表函数*}

<**select id="pageSize" name="dynamic-table\_length" aria-controls="dynamic-table"  
 class="form-control input-sm"**>  
 <**option value="10"**>10</**option**>  
 <**option value="25"**>25</**option**>  
 <**option value="50"**>50</**option**>  
 <**option value="100"**>100</**option**>  
</**select**> 条记录 </**label**>

*//带分页参数的ajax请求***function** *loadUserList*(deptId) {  
 **var** pageSize = **$**(**"#pageSize"**).val();  
 **var** url = **"/sys/user/page.json?deptId="** + deptId;  
 **var** pageNo = **$**(**"#userPage .pageNo"**).val() || 1; *//第一次请求时pageNo数值为1* **$**.ajax({  
 **url**: url,  
 **data**: {  
 **pageSize**: pageSize, *//分页参数* **pageNo**: pageNo *//分页参数* },  
 success: **function** (result) {  
 *renderUserListAndPage*(result, url);  
 }  
 })  
}

ajax请求路径：

@RequestMapping(**"/page.json"**)  
@ResponseBody  
**public** JsonData page(@RequestParam(**"deptId"**) **int** deptId, PageQuery pageQuery) {  
 PageResult<SysUser> result = **sysUserService**.getPageByDeptId(deptId, pageQuery);  
 **return** JsonData.*success*(result);  
}

传入分页实体类PageQuery、返回分页实体类结果PageResult：