EasyUi使用

在把文件夹复制在工程目录下

1. jsp页面<head>标签体引入

<link rel="stylesheet" type="text/css" href="jquery-easyui-1.3.3/themes/default/easyui.css">

<link rel="stylesheet" type="text/css" href="jquery-easyui-1.3.3/themes/icon.css">

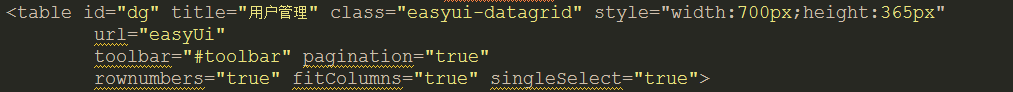
<script type="text/javascript" src="jquery-easyui-1.3.3/jquery.min.js"></script>

<script type="text/javascript" src="jquery-easyui-1.3.3/jquery.easyui.min.js"></script>

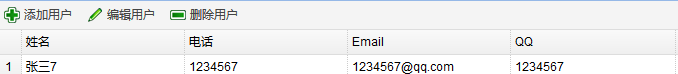
<script type="text/javascript" src="jquery-easyui-1.3.3/locale/easyui-lang-zh\_CN.js"></script>

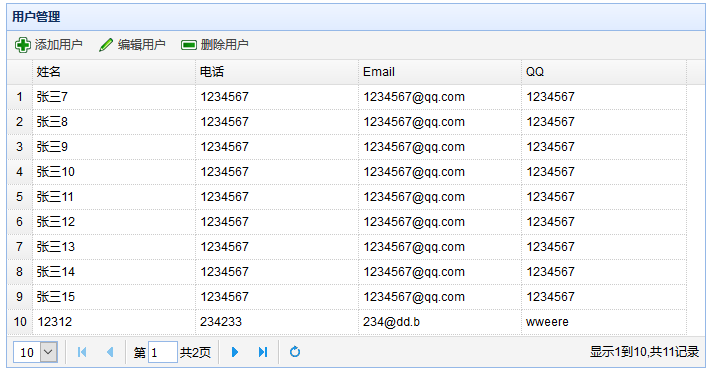
2.

<!-- url:servlet映射 pagination: 页数属性 rownumbers: 行数属性 singleSelect： 单选属性 plain:button使用时可以使得按钮样式消失，跟背景融合一起-->



对应显示的样式



例子：[H:\\code\\EasyUi\\WebContent\\testOne.jsp](file:///H:\code\EasyUi\WebContent\testOne.jsp)

3. <!-- editable="false":不可编辑 panelHeight="auto":控件高度自适应 class="easyui-combobox":选择框 class="easyui-datebox":日期框 data-option:用来装载easyui的各种属性 -->

4、在table 中使用数据表格时，class="easyui-datagrid".

<th>头部的<!-- field的名称要跟数据库定义的字段名称一致 -->

5、如果从数据库获取的数据是日期格式转换成josn格式时要先把日期格式转为字符串格式

6.添加自己想使用的icon图标，把自己喜欢的图标收集起来放到一个文件夹中，把文件夹放到easyui开发包的theme目录下，然后在icon.css中添加使用的类名以及源地址

Easyui通过数据库生成树

数据库的表的设计必须为生成树的属性

Html代码

<ul class="easyui-tree" id="tree"></ul>

Jquery代码获取后台数据

$(function(){

$("#tree").tree({

url:'auth?action=menu&parentId=-1',

lines:true,

onLoadSuccess:function(){

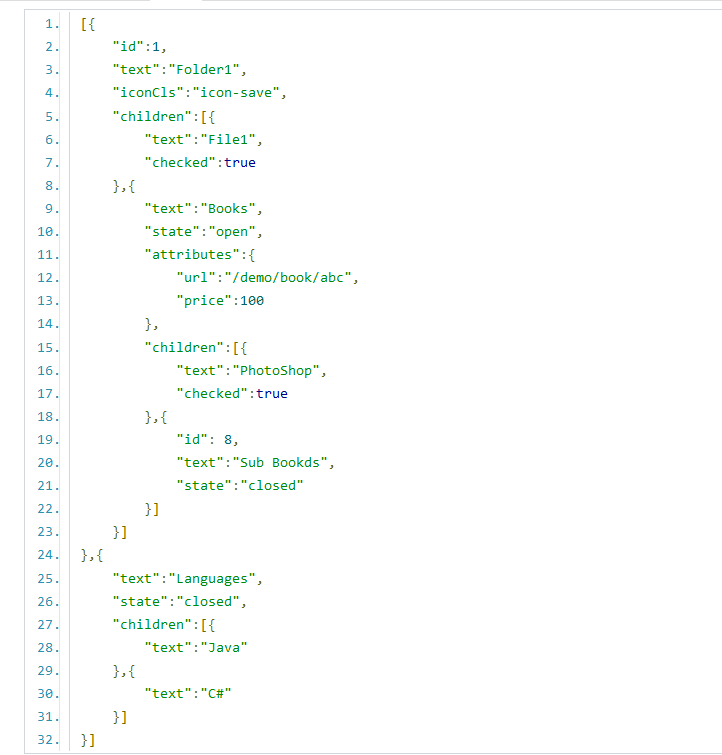
$("#tree").tree('expandAll');

}

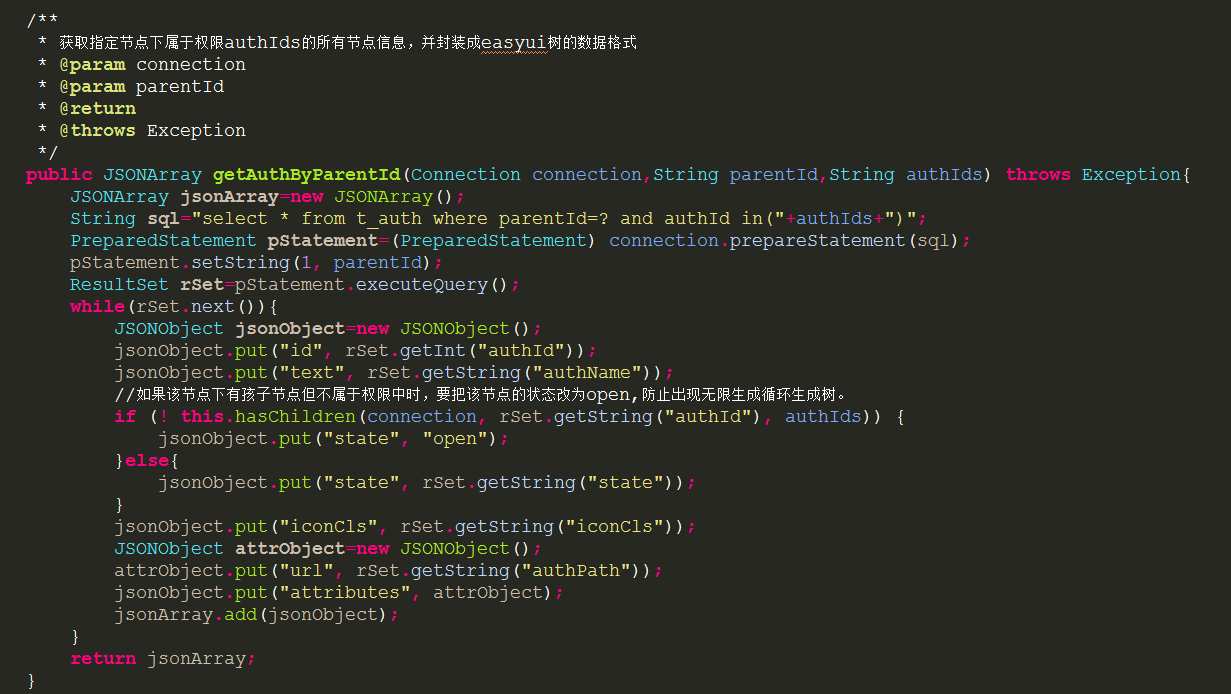
});

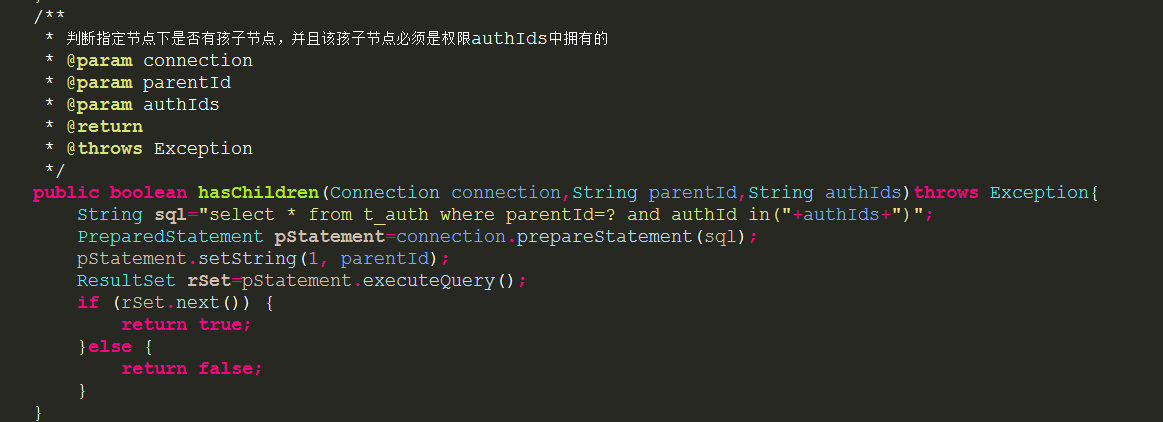
})

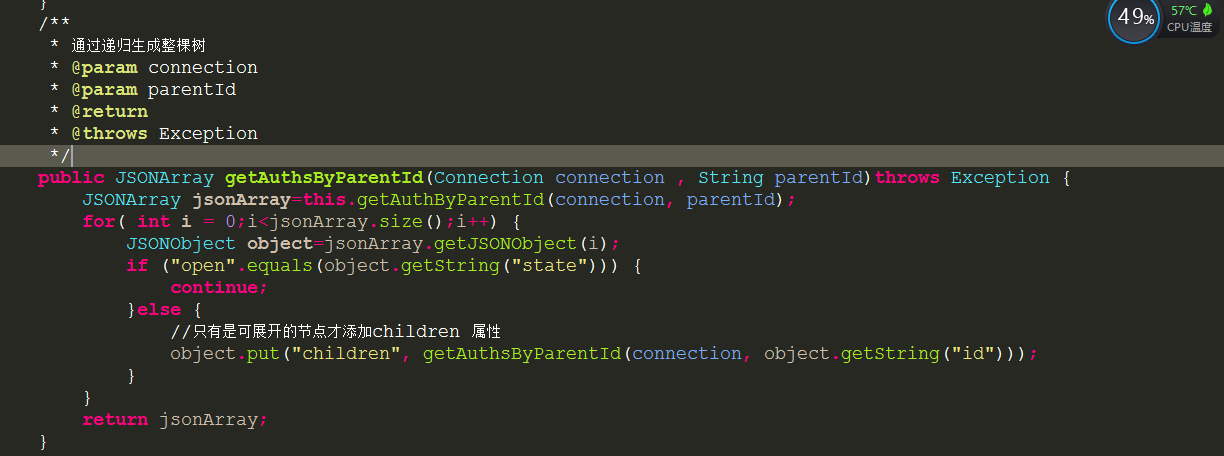
因为生成树的数据格式为：



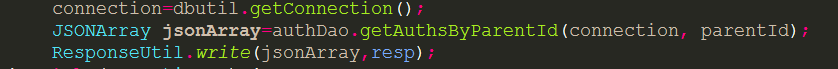
所以后台需要将获取到的数据封装成相应的格式，在获取权限树时有一个注意点是，如果有一个节点为父亲节点，该节点下有子节点存在，但其不是该角色的权限时，需要把该父亲节点的state改为open，否则会出现逻辑bug，树会一直在无限的生长下去







最后将数据返回页面



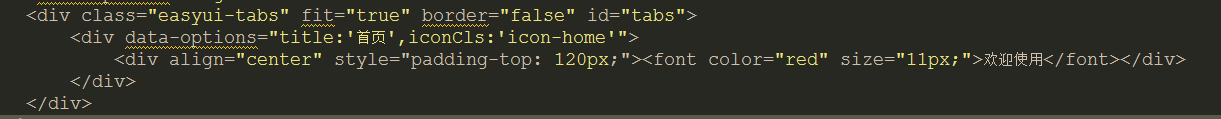
整体布局：

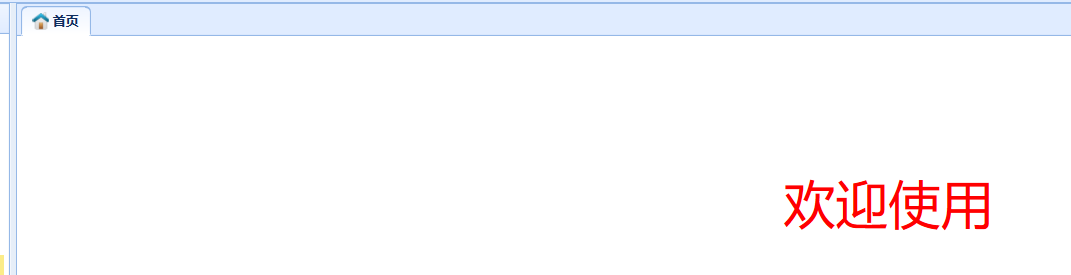


主要使用easyui-layout做整体，region=”north”, region=”center”, region=”west”, region=”south”分块，split控制是否可滑动，通过style设置宽度或者高度控制大小

region=”center”一般是整体显示功能区，

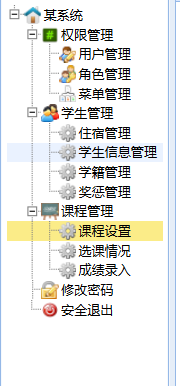
功能区通过添加class="easyui-tabs"



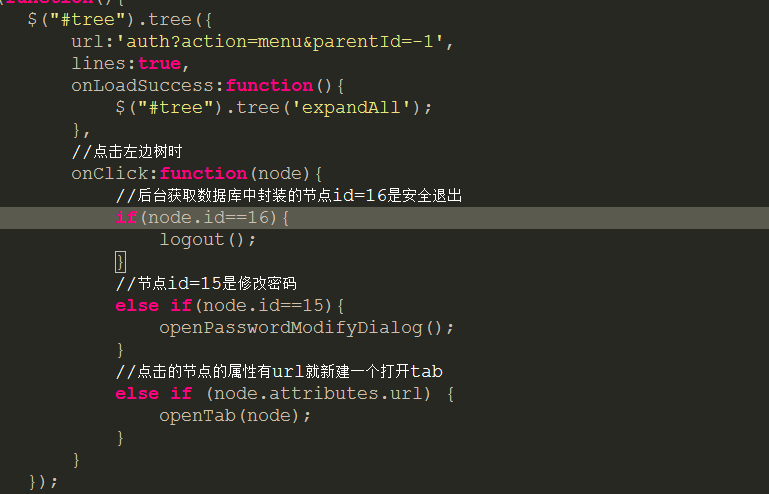
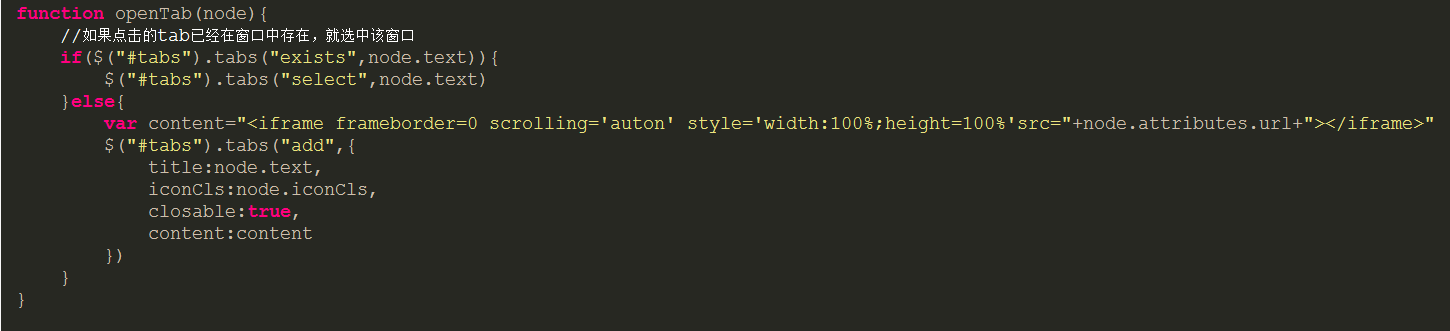


region=”south”，一般为底部

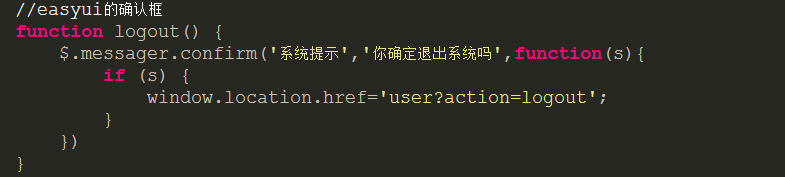
region=”west”,用来显示树，树生成的需要的数据通过后台获取

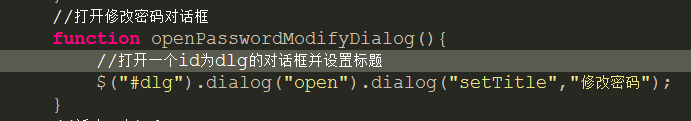


点击树新建相应的窗口

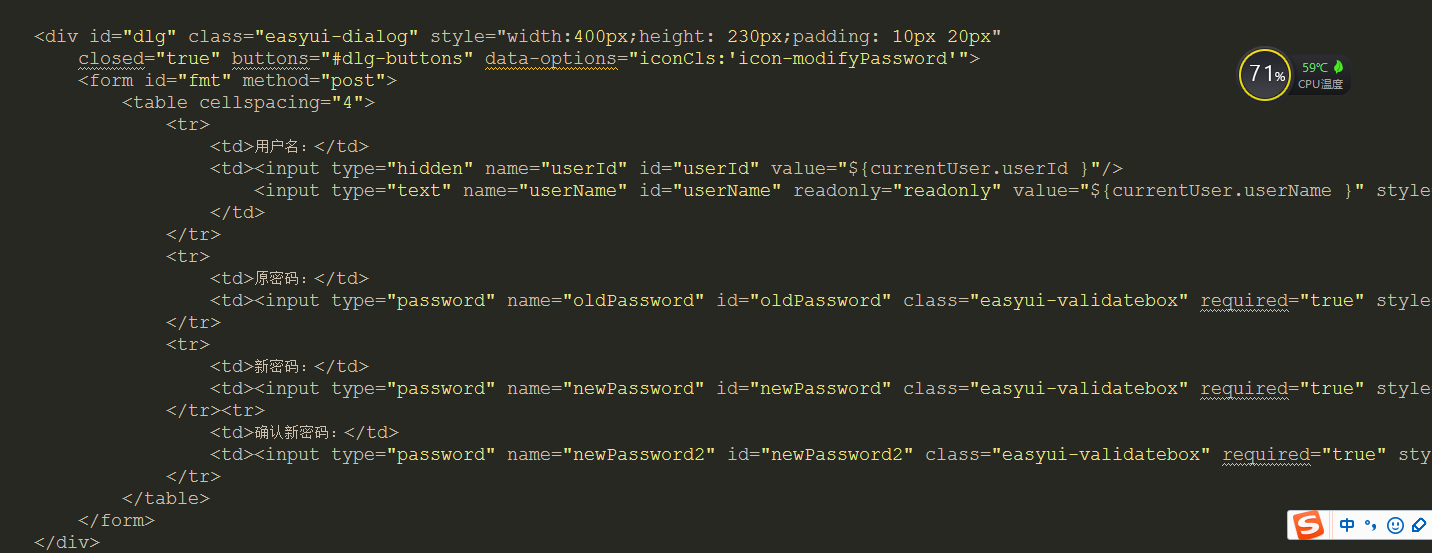
 

安全退出使用easyui确认框

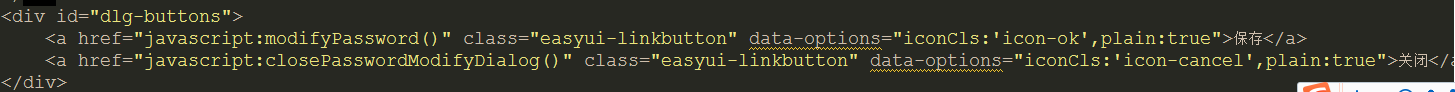




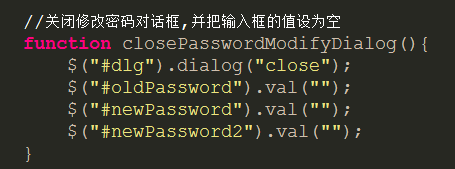
自己建立的对话框



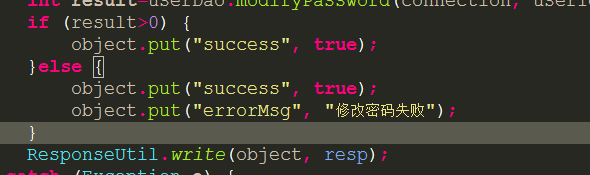
buttons="#dlg-buttons" 添加指定id的button，并给添加相应的点击事件





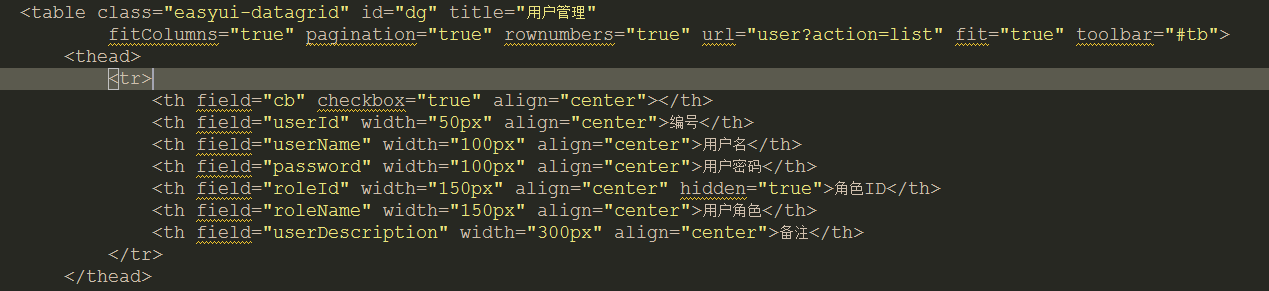


修改密码后天只是做相应的对数据库的操作，几返回结果给页面



创建用户管理页面

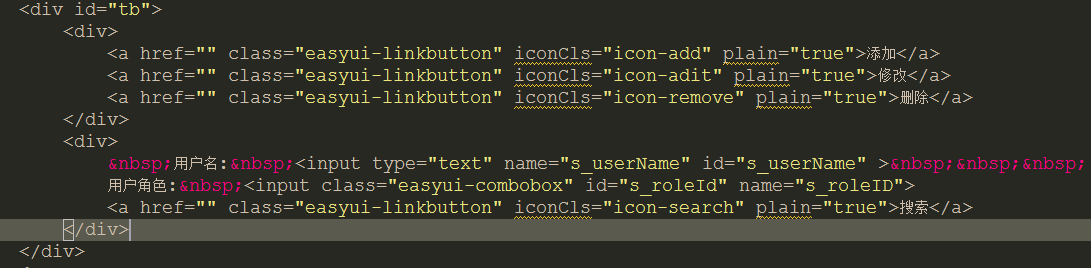
使用datagrid



Title是表格名称 fitColumns:行自适应 pagination:显示页数 rownumbers:显示行数 url:加载表格时自动访问指定url获取数据，fit：自适应大小

后台返回的数据格式：

Toolbar:加载指定id的toolbar 一般都是linkbutton



显示

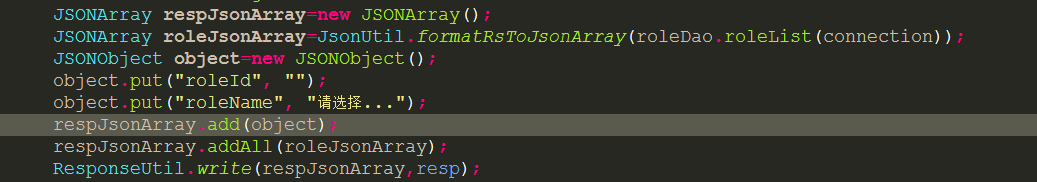


实现combobx从后台获取数据填充下拉列表



添加data-options，url：获取数据的地址 textField：显示在页面的值，后台数据中的key，panelHeight：自适应高度 editable:是否可编辑

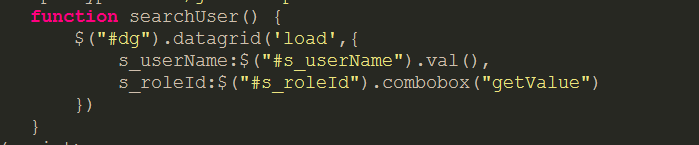
后台返回的格式为JSONArray,第一个时手动添加的只是为了提高用户体验





实现搜索 添加js函数

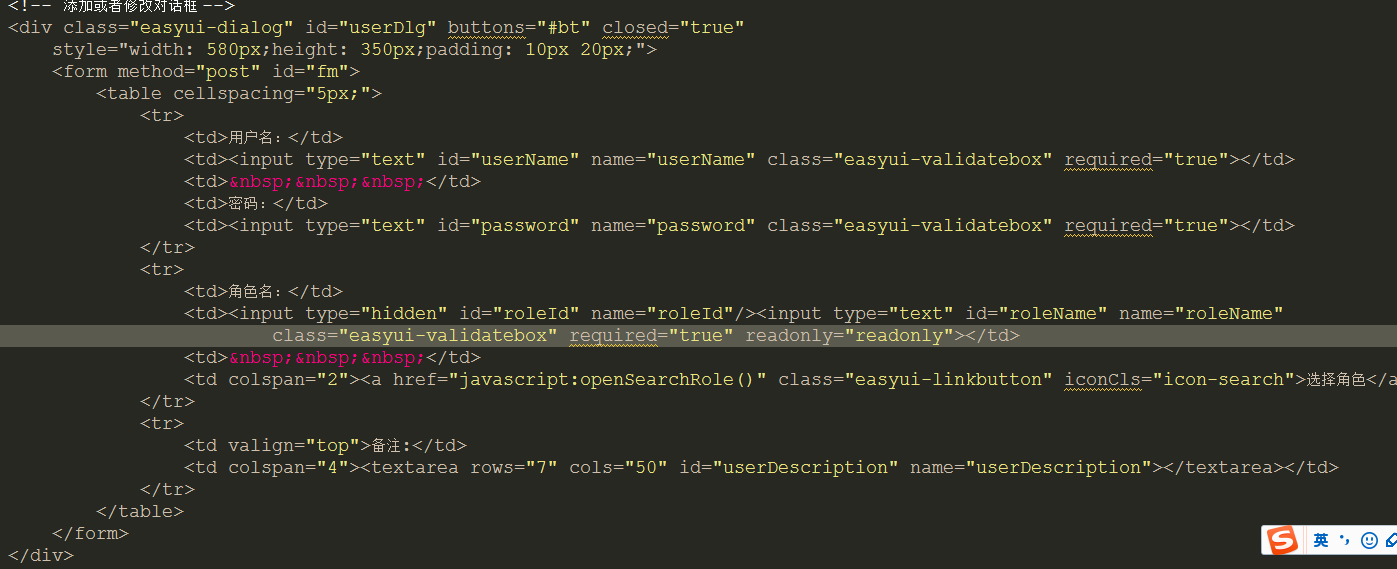




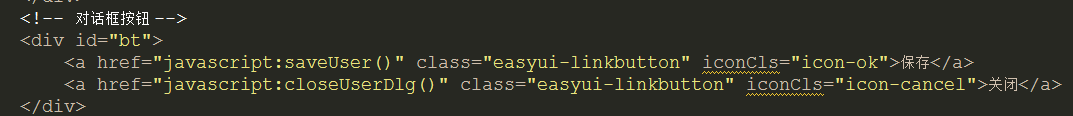
该函数是选取id为dg的数字表格通过数据表格自带的load方法并携带两个K-V形式的参数到id为dg指定的url,获取返回的数据会在id为dg的表格中展示。

实现添加功能

编写添加或者是修改时的对话框：



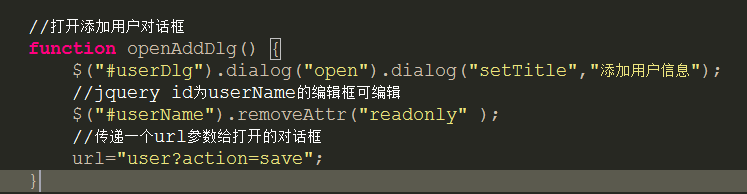
添加连个底部按钮

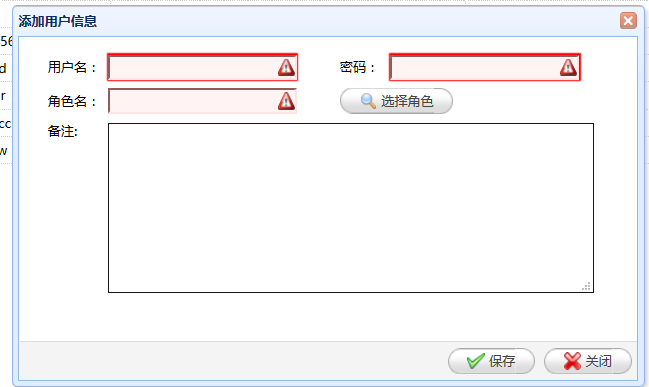


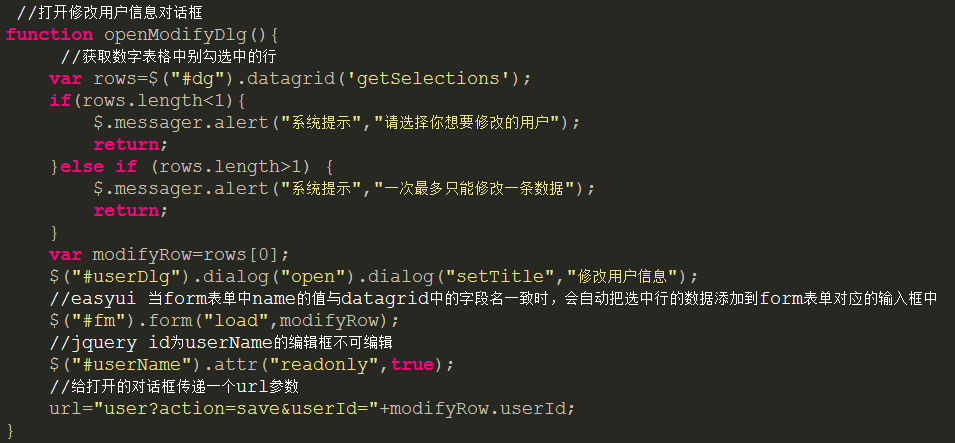
在toolbar的按钮中添加点击事件

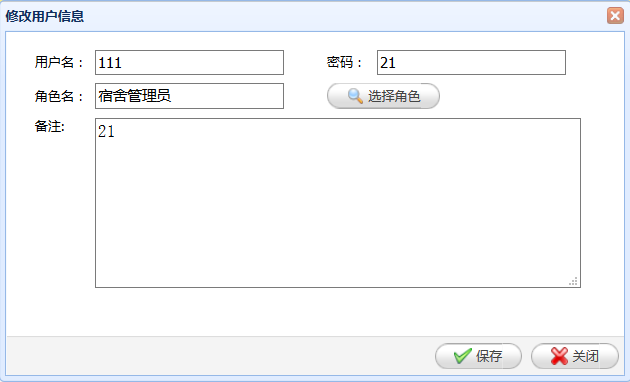


通过js打开添加用户对话框或者修改用户信息对话框，添加用户于修改用户打开的都是id=userDlg的对话框，只是修改对话框需要加多一点相应的属性设置







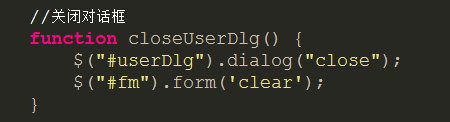


然后为对话框底部的两个按钮添加点击事件：



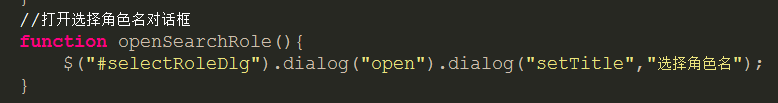


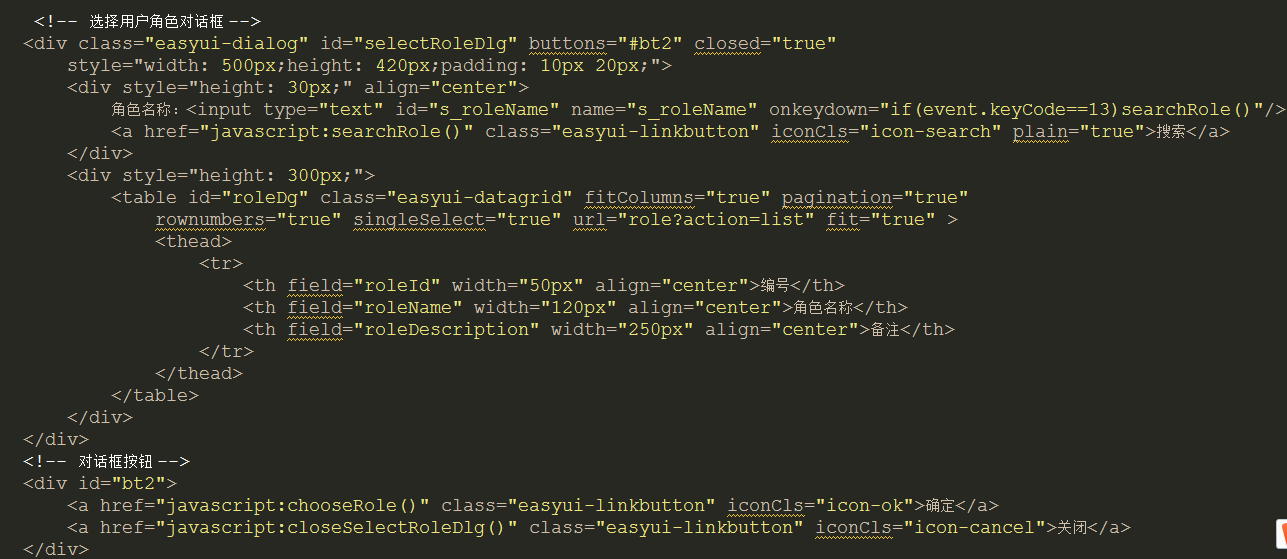
因为添加或者保存使用的都是同一个对话框，因此点击保存时触发的是同一事件，为了区分添加以及保存的url的不同，使用全部变量，并且在相应打开对话框时为该url附上值



为了确保角色是数据库中存在的，所以得限定角色只能是选取



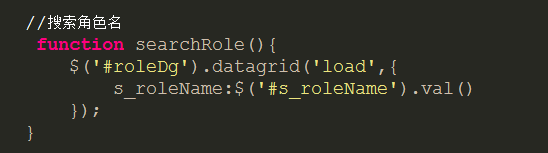




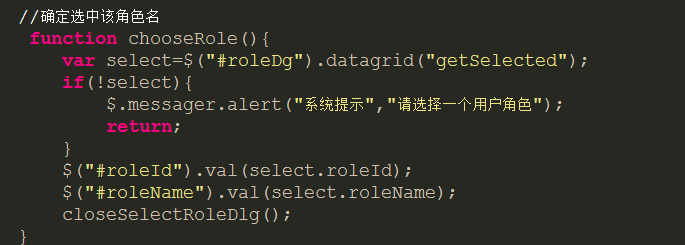
从给定的url获取返回数据，然后，生成表格供用户选择，需要的是后台返回的数据是Json格式，所以需要见查询到的resultSet转为json格式，封装好有一个util



当用户角色足够多是，所以需要搜索功能,当执行该函数的时候，会重新加载指定id的datagrid的url,并携带传递的参数，获取返回结果，重新刷新页面数据



当用户确定选中的角色时，往上一个页面的填值



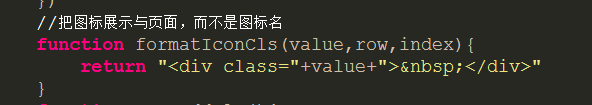


编写菜单管理页面：

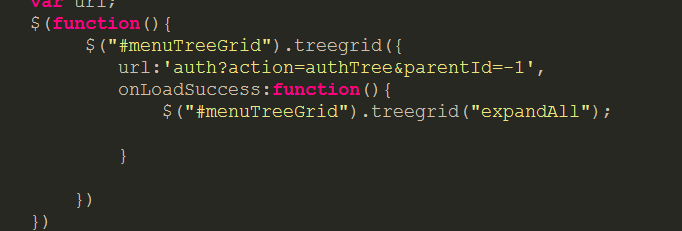
页面使用树形表格treegrid,格式跟表格的使用差不多，data-option是一些显示的属性设置



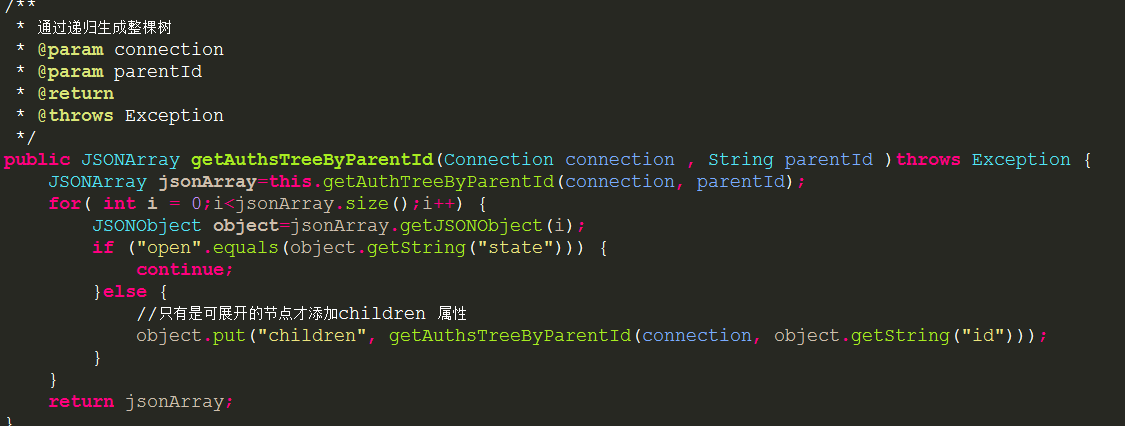
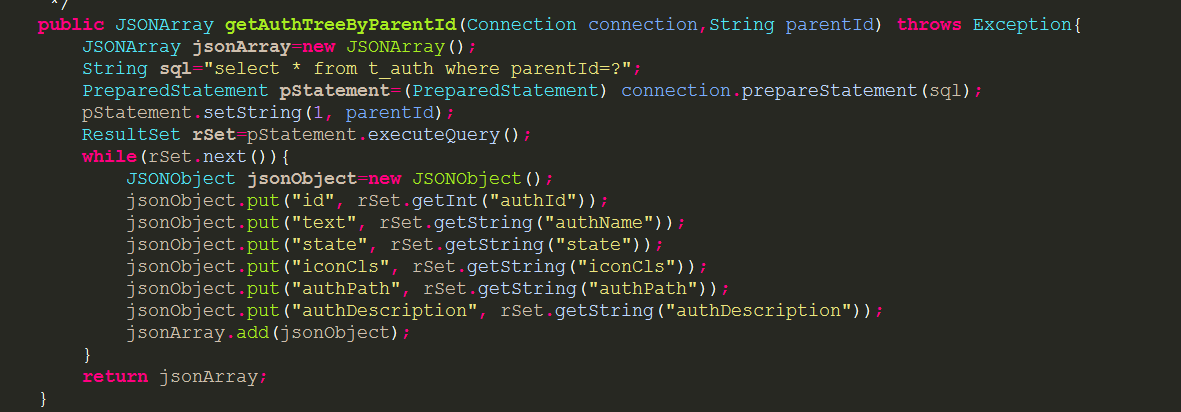
注意的是图标的展示的是样式，而不是名字时，要使用formatter函数，然后添加一个自己写的js函数



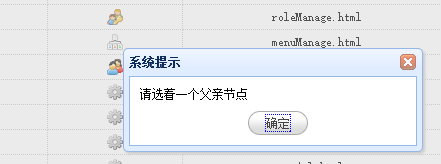
通过写js加载后台数据，使用的是页面加载完毕自动执行



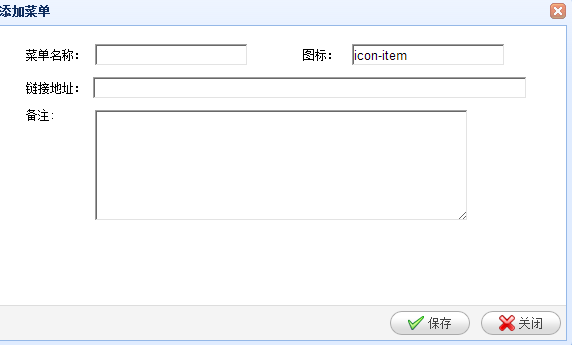
后台封装的从数据库获取的数据代码：



添加菜单时，根据需要先要用户选取在哪添加，



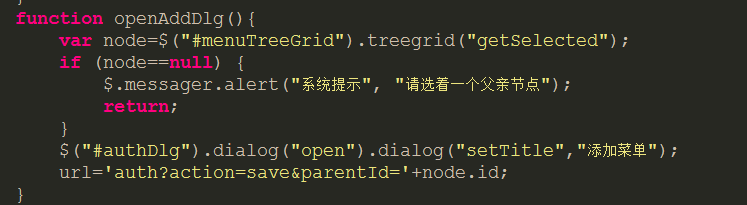
添加页面

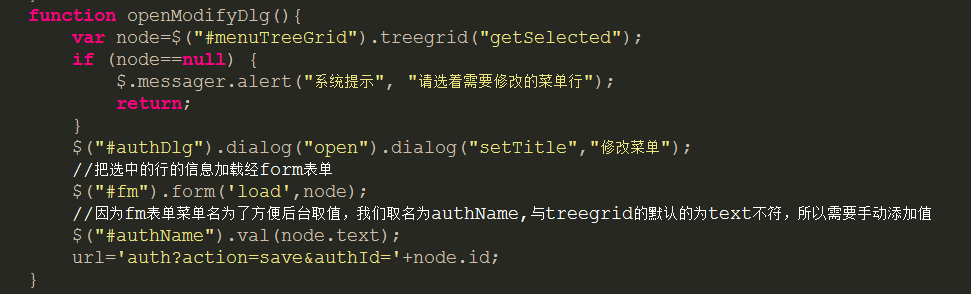


页面代码，图标的样式，设一个默认值

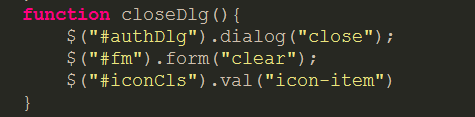


修改跟添加使用的是同一个页面框，，不一样的是修改时把选到的数据填充到页面中

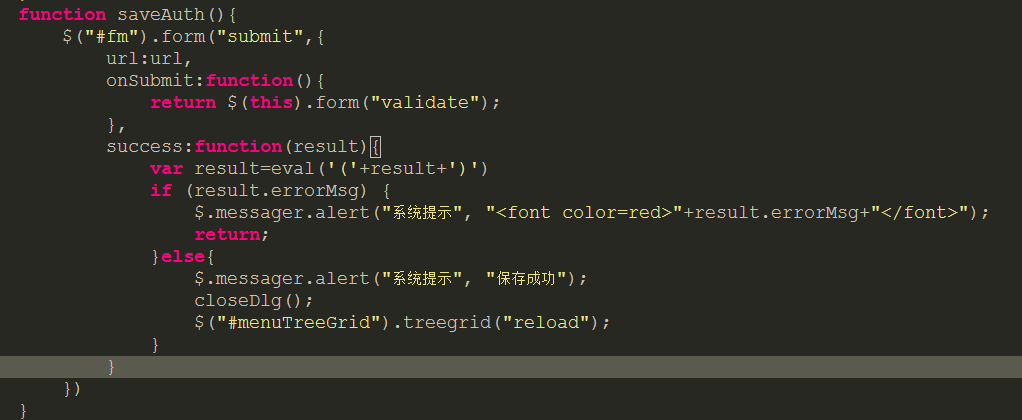




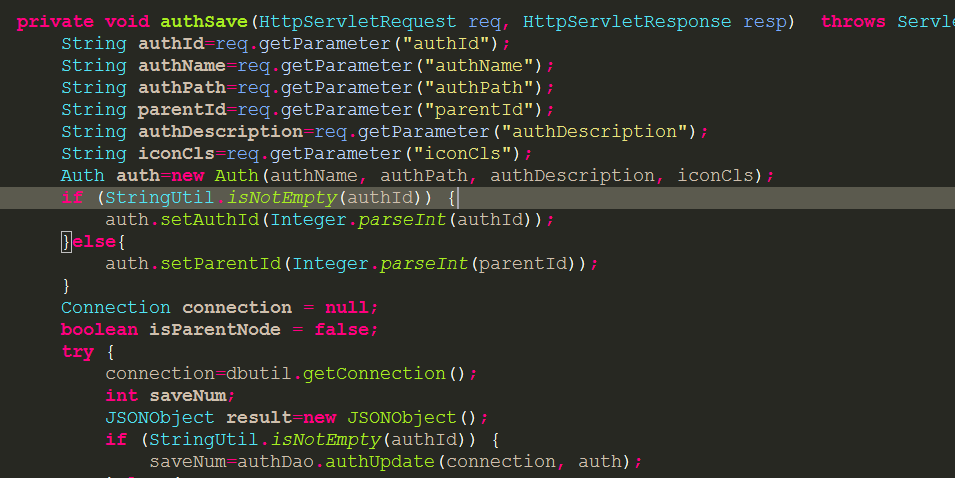
然后为页面的保存或者关闭添加点击事件



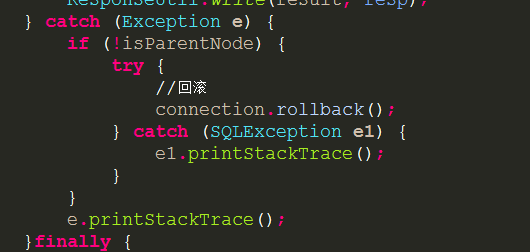
因为页面使用的是form表单，保存时使用form表单的提交格式

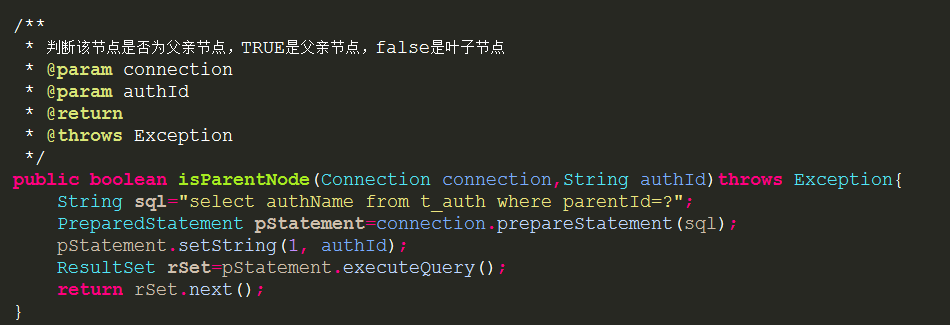
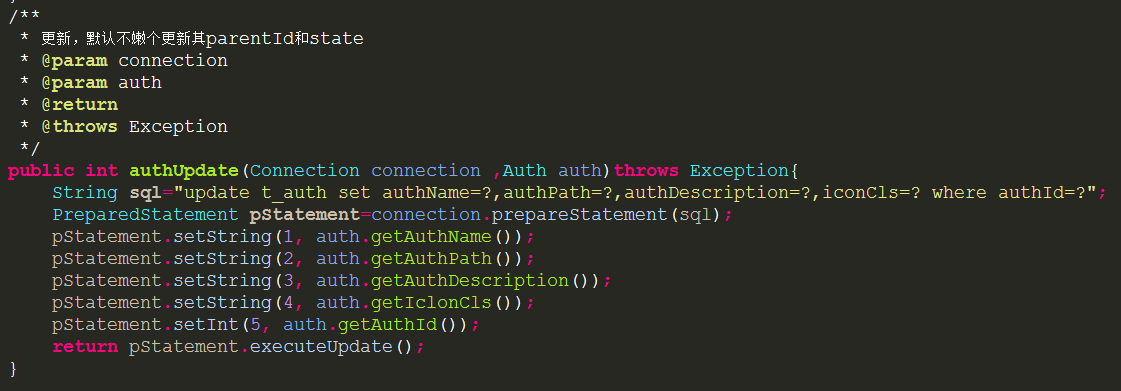
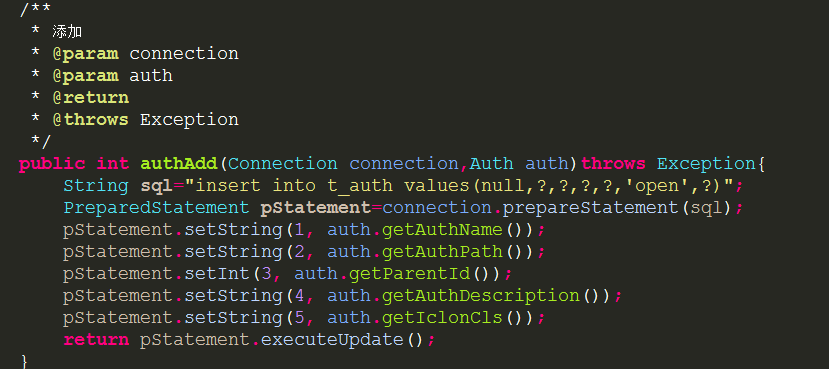


后台处理时需要注意一个就是判断用户的动作是添加还是修改，是添加时有一个注意事项就是用户选着添加的节点原来是叶子节点，还是父亲节点，如果原来是叶子节点的话还需要将该节点的状态state改为closed，确保获取树时可以保证树的完整性，通过authId判断是添加还是修改，authId为空就是添加，不为空时修改，当是添加操作时需要parentId,因此需要前台在访问url时传递parentId的值，还有另一个注意事项是，在进行添加操作时，当涉及到更改父节点的state，如果程序抛出异常时，要确保事务回滚，所以要使用事务提交







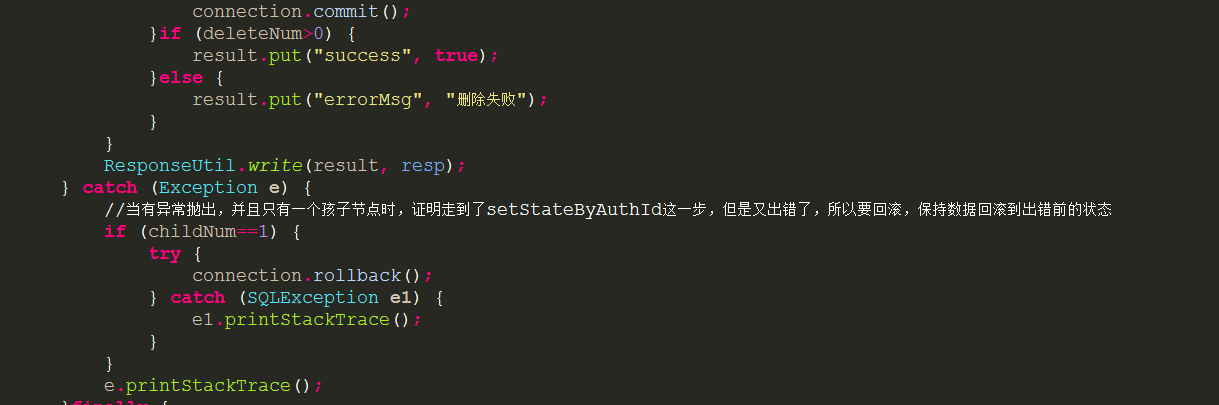
对数据库的操作代码：

判断是否为父亲节点，在进行添加时需要使用

用户删除操作

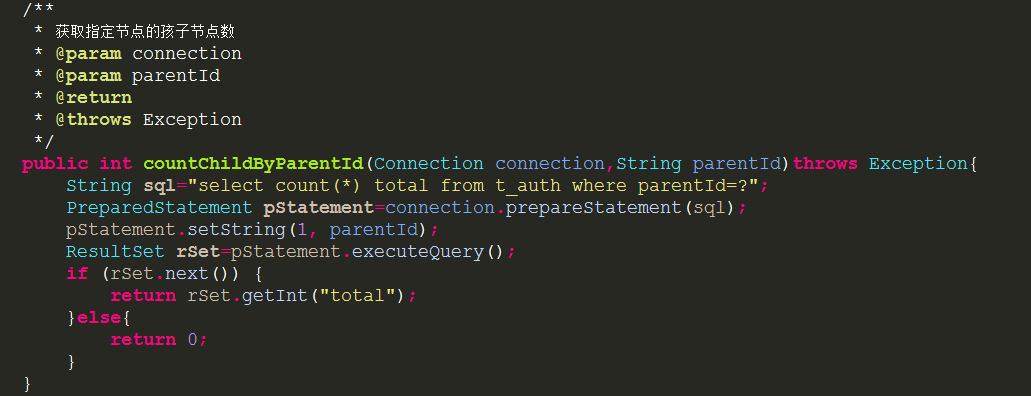
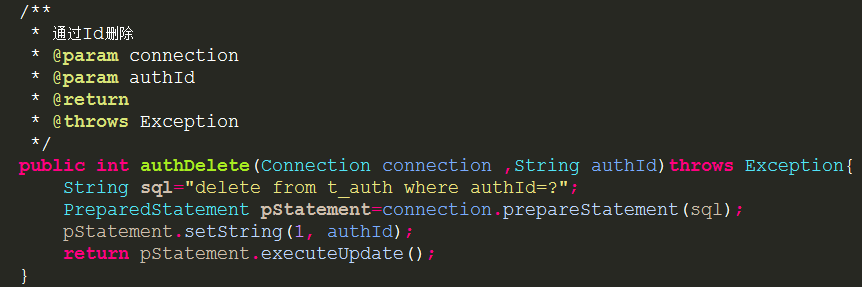
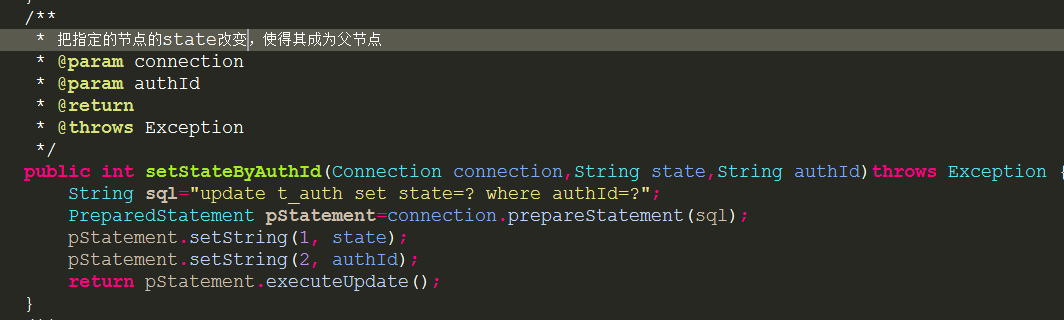


当用户进行删除时，需要考虑有三种情况

1. 删除的是叶子节点，删除该节点之后，其还有兄弟节点存在，正常删除
2. 删除的是叶子节点，删除该节点之后，其没有兄弟节点存在，需要把其父亲节点的state改为open
3. 删除的是父亲节点，其下边存在着叶子节点，无法删除
4. 

这里修改父亲的state时也需要到使用事务提交，并在抛出异常时判断以及回滚事务

对数据库的操作代码



对于判断是否是父亲节点时，在添加操作时已有代码