神宁安全风险预控系统项目总结

一.echart报表

总结：

1）准备部分

1.如何生成树：

```
    //生成部门树形菜单
 public String tree() throws IOException{
        HttpServletResponse response=ServletActionContext.getResponse();
            response.setContentType("text/html");
            response.setContentType("text/plain; charset=utf-8");
            PrintWriter out;
            out = response.getWriter();
            String hql="";
        //判断id是否为空或者为空字符串（点击节点时前台传过来节点id的值被自动命名为id）
            if(id!=null&&id.trim().length()>0){
             departments=departmentService.getDepartmentsByParentDepartmentSn(id);
            }
            else{
             hql = "FROM Department d where d.departmentSn= null";
             departments=departmentService.queryByHql(hql);
            }
  JSONArray tree=new JSONArray();
  for(Department department:departments){
                //循环一次创建一个新的json对象，不断往json集合里添加
   JSONObject jo=new JSONObject();
                //前台combotree只需要url，id和text会自动对应接收
    jo.put("id",department.getDepartmentSn());
    jo.put("text",department.getDepartmentName());
    if(department.getChildDepartments().size()>0){
     jo.put("state","closed");
    }
    else{
     jo.put("state","open");
    }
    tree.put(jo);
  }
  out.println(tree.toString());
              out.flush();
         out.close();
  return SUCCESS;
 }
```

2.如何动态生成按钮

```
    //加载部门类型
```

Action：

```
public String type() throws IOException{
      HttpServletResponse response=ServletActionContext.getResponse();
            response.setContentType("text/html");
            response.setContentType("text/plain; charset=utf-8");
            PrintWriter out;
            out = response.getWriter();
            JSONArray data=new JSONArray();
   departmentTypes=departmentTypeService.getImplDepartmentTypesExceptSelf(departmentSn);
 for(DepartmentType departmentType:departmentTypes){
  JSONObject jo=new JSONObject();
  jo.put("text",departmentType.getDepartmentTypeName());
  jo.put("value",departmentType.getDepartmentTypeSn());
  data.put(jo);
 }
 out.print(data.toString());
            out.flush();
            out.close();
 return SUCCESS;
}
```

jsp:

```
$.post("${pageContext.request.contextPath}/report/typeReport",{departmentSn:node.id},function(data){
    $("#typebtns").empty();
    $("#typebtns").append("<label>"+"部门类型：+"</label>");
    for (var i = 0;i < rv.length;i++){
      $("#typebtns").append("<a class='btn btn-default'  data='" + data[i].value+"'>"+ data[i].text+"
</a>");
    }
    $.parser.parse($('#typebtns').parent());
    <!--部门类型按钮点击 -->
    $("#typebtns a").click(function(){
     //点击取消
        {
        else
        {
            //点击选中
     departmentTypeSn=$(this).attr("data");
          }
    });
   },'json');
 }
```

3.时间：

（1）获得当前时间：

```
var date=new Date();
var endtime=date.getFullYear()+"-0"+(date.getMonth()+1)+"-"+date.getDate();
```

```
 var begintime=date.getFullYear()+"-0"+(date.getMonth())+"-"+date.getDate();
```

（2）遇见local时间类型的，在action里用date接受，然后用LocalDate localDate= date.toInstant().atZone(ZoneId.systemDefault()).toLocalDate()转化

（3）格式化字符串

```
SimpleDateFormat df = new SimpleDateFormat("yyyy-MM-dd：hh-mm-ss");
 str=df.format(new Date());
    //转成timestamp
    Timestamp.valueOf(str);
```

（4）使用周日历

引用js文件以及datepicker相关文件：

```
<link rel="stylesheet" type="text/css" media="screen" href="${pageContext.request.contextPath}/css/datePicker.css">
<script type="text/javascript" src="${pageContext.request.contextPath}/js/jquery.datePicker.js"></script>
```

设置选择时为周：在页面加载完成事件里加$('.date-pick').datePicker({selectWeek:true,closeOnSelect:true});

获得周日历的值，采用change事件：

```
$('#weekdate').change(function(){
   var time=$(this).val().toString();//根据需要再做处理
  });
```

2）echart相关部分

1.在body里新建echart控件

```
<div id="main" style="width: 100%;">
```

2.在页面加载完成事件里读取main的dom对象并初始化静态部分，显示标题，图例和空的坐标轴

```
var myChart = echarts.init(document.getElementById('main'));
myChart.setOption({
 tooltip : {
  trigger: 'axis',//鼠标移动到柱子上显示阴影
   axisPointer : {            // 坐标轴指示器，坐标轴触发有效
     type : 'shadow'         // 默认为直线，可选为：'line' | 'shadow'
  }
   },
  title: {
  text: '隐患报表',//大标题
  subtext: '直方图分析展示',//小标题
   x: 'center'
 },
 legend: {
    data:['部门']
 },
 xAxis: {
       data: []
  },
  yAxis: {},
  series: [{
```

```
markLine : {
 lineStyle: {
    normal: {
      type: 'dashed'//图例的指示线
       }
},
  data : [
       [{type : 'min'}, {type : 'max'}]//线从最低点连接到最高点
 ]
},
name: '不符合项',
type: 'bar', barWidth: '50'
}]
});
```

3.如何实现在echart上单击右键生成菜单

(1)在页面加载完成事件里禁掉浏览器默认的右键：

```
$(document).bind("contextmenu", function(){ return false; });
```

初始化右键菜单：menu = $.ligerMenu({

```
top: 20,
left: 100,
width: 120,
items:[ { text: '部门', click: loadechart},
      { text: '专业', click: loadechart},
      { text: '危险等级 ', click: loadechart} ,
{ text: '查看明细 ', click: loadechart} ]//可以动态拼字符串
  });
```

右键单击事件： myChart.on('mousedown', function (params) {

```
if (params.event.event.which == 3) {}//代表右键单击
   }) ;
```

右键选择菜单：if(event.srcElement.innerText!=null){}

得到所选择的柱子编号：params.dataIndex；

在查询函数里的回调函数中动态为echart填入数据：

```
myChart.setOption({
xAxis: {
data: data.departmentName,
            //rotate倾斜度（当横坐标太多难以展示完整时）
axisLabel: {rotate: 25, show: true, interval: 0}
},
series: [{
    name: '按照部门不符合项',
    data:  data.inconformityItemCount
}]
});
```

3）后台部分

当按几种标准进行查询时（遍历此标准并携带其他条件）最好在action里抽出一个通用函数，当遍历其中一个标准时再

调用此函数，传入所有参数。再在这个query函数里（此时遍历到了当前标准里，即此参数的值已经确定，与前台传回此参数无异）通过判断条件是否存在然后拼接hql语句并调用底层方法。进行查询。

二：easyui部分

1.datagrid:不设置宽度高度，设置fitcolumns和fit使其自适应。如果是在查询页面，在onBeforeLoad事件里，等查询条件都有值了返回true再放松请求或者不要设置url而在查询函数里load之前先给url赋值。

2.在datagrid里加toolbar,如果加入控件，先在页面新建:

```
<input id="treehidden" type="hidden">
```

再在datagridtoolbar里引用:

```
toolbar:[
   {
    text:'事故部门：<input id="treeselect" name="treehidden">'
   }
]
```

3.combogrid：

```
//内容改变时搜索数据
delay:200,
   mode:"remote"
```

三.动态生成表格

主要思路：

1.可以直接在脚本里写方法得到数据并动态生成表格内容

2.在action里得到数据，前台接受数据并遍历对应的集合动态生成表格

第一种尤其麻烦，并且效果不好，因为每次都是全部刷新，体验非常不好，看似代码少了，实则增加了大麻烦。

第二种步骤：

1.在后台取得几个集合，包括横纵坐标标题以及编号（用来单击单元格时查看明细取得数据时所用），还有就是单元格的数据（一般是数量）。

action方法：

```
public String report() throws IOException{
  out();
  List<Department> departments=departmentService.getDepartments(departmentSn, departmentTypeSn);
  List<NearMissType> nearMissTypes=nearMissTypeService.getAllNearMissType();
  JSONArray array=new JSONArray();
  for(Department de:departments){
   JSONObject jo=new JSONObject();
   jo.put("deptName", de.getDepartmentName());
   jo.put("deptSn", de.getDepartmentSn());
   JSONArray types=new JSONArray();
   for(NearMissType type:nearMissTypes){
    JSONObject js=new JSONObject();
    String hql="select count(n) from NearMiss n where n.department.departmentSn like
'"+de.getDepartmentSn()+"%' and n.nearMissType.nearMissTypeSn="+type.getNearMissTypeSn()+" and
n.happenDate between '"+begintime+"' and '"+endtime+"'";
    js.put("typeCount", nearMissService.countHql(hql));
    js.put("typeName", type.getNearMissTypeName());
    js.put("typeSn", type.getNearMissTypeSn());
```

```
            types.put(js);
           }
           jo.put("types", types);
           JSONArray levels=new JSONArray();
            for(int i=0;i<3;i++){
             JSONObject jl=new JSONObject();
            String hql="select count(n) from NearMiss n where n.department.departmentSn like
'"+de.getDepartmentSn()+"%'and n.riskLevel="+i+" and n.happenDate between '"+begintime+"' and
'"+endtime+"'";
             jl.put("levelCount", nearMissService.countHql(hql));
             jl.put("levelSn", i);
             levels.put(jl);
           }
           jo.put("levels", levels);
           array.put(jo);
          }
          String str="{\"array\":"+array+"}";
          out().print(str);
          out().flush();
          out().close();
          return SUCCESS;
         }
```

2.前台生成表格的函数：

```
function loadtable(){
 $.post("${pageContext.request.contextPath}/attempted/event/reportNearMiss",
{departmentSn:departmentSn,departmentTypeSn:departmentTypeSn,begintime:begintime,endtime:endtime},fun
ction(str){
  $("#table").empty();
  $("#table").append("<tr><td>序号</td><td>单位</td><td style='text-align:center;'
colspan="+str.array[0].types.length+">事件类别</td><td style='text-align:center;' colspan='3'>风险等
级</td></tr>");
  $("#table").append("<tr>");
  $("#table").append("<td colspan='2'></td>");
  for(var i=0;i<str.array[0].types.length;i++){
   $("#table").append("<td>"+str.array[0].types[i].typeName+"</td>");
  }
  $("#table").append("<td>一般风险</td><td>中等风险</td><td>严重风险</td>");
  $("#table").append("</tr>");
  for(var i=0;i<str.array.length;i++){
   $("#table").append("<tr>");
   $("#table").append("<td>"+i+"</td>");
   $("#table").append("<td>"+str.array[i].deptName+"</td>");
   for(var j=0;j<str.array[i].types.length;j++){
    var other=null;
    //$("#table").append("<td>"+str.array[i].types[j].typeCount+"</td>");
```

```
        $("#table").append("<td><a  href='#'
    onClick='de("+str.array[i].deptSn+","+str.array[i].types[j].typeSn+","+other+")'>"+str.array[i].types
    [j].typeCount +"</a></td>");
       }
       for(var g=0;g<str.array[i].levels.length;g++){
        //$("#table").append("<td>"+str.array[i].levels[g].levelCount+"</td>");
        var other=null;
        $("#table").append("<td><a  href='#'
    onClick='de("+str.array[i].deptSn+","+other+","+str.array[i].levels[g].levelSn+")'>"+str.array[i].lev
    els[g].levelCount+"</a></td>");

       }
       $("#table").append("</tr>");
      }
     },'json');
    }
```

四.poi

1.导出excel时的struts配置:

```
<action name="export" class="" method="export">
   <result type="stream">
  <param name="contentType">application/octet-stream(vnd.ms-excel)</param>
               <param name="inputName">wordStream（读取action里的输出流）</param>
         <param name="contentDisposition">attachment;filename=${wordFileName}</param>
               <param name="bufferSize">4096</param>
           </result>
</action>
```

2.导入excel数据：

1)action

读取文件并转化为输入流：

```
     InputStream is = new FileInputStream(excel);
     XSSFWorkbook wb = new XSSFWorkbook(is);
```

设置字符串变量记录错误：nullData+=(rowNum+1)+",";continue;

读取excel里的数字或者字符串并转换为字符串：

```
//转换并读取字符
 public String StringValue(XSSFCell before){
  String after="";
        if(before.getCellType()==Cell.CELL_TYPE_STRING){
         after=before.toString();
        }
        if(before.getCellType()==Cell.CELL_TYPE_NUMERIC){
         BigDecimal bd = new BigDecimal(before.getNumericCellValue());
         after=bd.toPlainString();
        }
        return after;
```

}

3.导出成word

主要思路：建立模板文件，动态替换数据

步骤：

```
//读取模板文件：String path
=ServletActionContext.getServletContext().getRealPath("/template/managementReview.doc");
//替换占位符：
Range range=doc.getRange();
range.replaceText("${1}",managementReview.getPurpose());
ByteArrayOutputStream fout = new ByteArrayOutputStream();
//输出
doc.write(fout);
 doc.close();
 fout.close();
 byte[] fileContent = fout.toByteArray();
 ByteArrayInputStream is = new ByteArrayInputStream(fileContent);
 //把doc输出到输出流
 wordStream=is;
wordFileName=URLEncoder.encode("评审报告.doc","UTF-8");
```

五.已解决问题：

1.Cannot call sendError () after the response had been commited.

原因： action里的返回值与struts的result不匹配，页面加载时发送了两次相同的请求，action里注入了service层，因为有set和get方法，而服务器当做变量输出。

解决办法：一味返回默认值SUCCESS比较方便美观,在tree的onLoadSuccess事件里删掉调用通过方法的代码，只需要在这里默认选中根节点，然后在tree的select事件里调用一次通过方法既可。

2.按钮点击变小

原因：bootstrap和easyui引用的样式文件冲突,<a class="easyui-linkbutton btn">

解决办法：删掉easyui-linkbutton

3.无法给变量赋值

原因：变量名与函数名重复，被认为是函数，所以无法赋值

提醒：变量名千万不能重复，也不能与函数名和关键字重复，

4.点击表单提交按钮无反应<input class="easyui-datebox easyui-envalidate">

原因：easyui-datebox easyui-envalidate冲突，datebox有两个框，有一个隐藏的框含有真正 的被提交的值，但是为空，表单验证看似正常，其实隐藏框为红色，未通过验证，所以无法提交

六.有用的小知识点：

1.查询和分页在一个函数里，复杂条件查询拼接函数可以抽出，查询参数拼接在一个函数里，一组按钮点击事件$(this)为当前点击的按钮

2.spring采用注解：只需在service和dao层加注解，在baseAction里用注解注入

3.$("#id").blur(function(){//离开时触发，可用于密码验证（密码验证可设置全局逻辑变量，如验证不通过则不能提交）；动态加样式：$("#oldpassTip").css({"display":"none"});设置延时：setTimeout(function(){};

4.分页加载datagrid时在action里根据条件拼接hql语句，两条语句几乎一样，只有一点不同，可替换，然后调用通过查询实体和数目的方法。

七.未解决问题：

1.为何通过struts配置输出hashmap很慢？

2.传递参数问题和日期问题，更简单