CRM 综合练习

1.1 客户管理 - 上传客户资质图片

1.1.1 文件上传回顾

1.1.1.1 什么是文件上传

- 将本地文件以流的形式写到服务器上

1.1.1.2 文件上传的技术

- JspSmartUpload

jspSmartUpload组件是应用JSP进行B/S程序开发过程中经常使用的上传下载组件，它使用简单，方便。现在我又为其加上了下载中文名字的文件的支持，真的是如虎添翼，必将赢得更多开发者的青睐。

- FileUpload

FileUpload 是 Apache commons下面的一个子项目，用来实现Java环境下面的文件上传功能，与常见的SmartUpload齐名。

- Servlet3.0
  - 文件上传
  - 注解开发
  - 异步请求
- Struts2 框架
  - 底层的实现:FileUpload,对 FileUpload 进行封装.

1.1.1.3 文件上传的要素

- 表单提交的方式必须是 POST
- 表单中需要提供 选择文件 未选择任何文件 ,而且这个文件项必须有 name 属性和值
- 表单的 enctype 属性必须是 multipart/form-data

1.1.2 文件上传代码实现

1.1.2.1 第一步:修改JSP页面(添加)

- 提供文件上传项

```
<TR>


    <td>客户资质：</td>
    <td colspan="3">
        <input type="file" name="upload" />
    </td>
</TR>
```

- 修改表单的 enctype 属性

```
<FORM id=form1 name=form1
    action="${pageContext.request.contextPath }/customer_save.action"
    method=post enctype="multipart/form-data">
```

1.1.2.2 第二部:修改 Action 中的 save 方法

- Struts2 的文件上传
    - 在 Action 中提供三个属性,对三个属性提供 set 方法
        - 字符串类型:　　上传项名称 + FileName
        - 文件类型:　　上传项名称
        - 字符串类型:　　上传项名称 + ContentType

```
/**
 * 文件上传提供的三个属性
 */
private String uploadFileName;      // 文件名称
private File upload;                // 上传的文件
private String uploadContextType;   // 文件类型

public void setUploadFileName(String uploadFileName) {
    this.uploadFileName = uploadFileName;
}

public void setUpload(File upload) {
    this.upload = upload;
}

public void setUploadContextType(String uploadContextType) {
    this.uploadContextType = uploadContextType;
}
```

```
    /**
     *
```

```
     * Title: save
     * Description: 客户管理:保存客户
     * @return
     * @throws IOException
     */
    public String save() throws IOException {
        //文件上传

        if(upload!=null) {
            //上传资质图片
            //设置文件上传的路径
            String path = "d:/upload";

            // 一个目录下存放的相同文件名:文件名随机
            String uuidFileName = UploadUtils.getUuidFileName(uploadFileName);
            // 一个目录下存放的文件过多:目录分离
            String realPath = UploadUtils.getPath(uuidFileName);

            //创建目录
            String url = path + realPath;
            File file = new File(url);
            if (!file.exists()) {
                //创建多级目录
                file.mkdirs();
            }

            //进行文件上传
            File destFile = new File(url + "/" + uuidFileName);
            FileUtils.copyFile(upload, destFile);
        }

        customerService.save(customer);
        return NONE;
    }
```

1.1.2.3 第三步:将文件上传的路径保存到数据库中

- 修改实体

```java
public class Customer {

    private Long cust_id;
    private String cust_name;
//  private String cust_source;
//  private String cust_industry;
//  private String cust_level;
    private String cust_phone;
    private String cust_mobile;
    private String cust_image;//文件上传的路径

    /*
     * 客户和字典是多对一关系,需要在多的一方放置一的一方的对象
     */
    private BaseDict baseDictSource;
    private BaseDict baseDictIndustry;
    private BaseDict baseDictLevel;

    public BaseDict getBaseDictSource() {
        return baseDictSource;
    }
```

- 修改映射

```xml
<hibernate-mapping>
    <class name="com.admiral.crm.domain.Customer" table="cst_customer">
        <!--建立类中的属性与表中的主键的映射关系 -->
        <id name="cust_id" column="cust_id">
            <!--主键的生成策略 -->
            <generator class="native"></generator>
        </id>

        <!-- 建立类中的普通属性与表中的字段的映射 -->
        <property name="cust_name" column="cust_name" />
        <!-- <property name="cust_source" column="cust_source" />
        <property name="cust_industry" column="cust_industry" />
        <property name="cust_level" column="cust_level" /> -->
        <property name="cust_phone" column="cust_phone" />
        <property name="cust_mobile" column="cust_mobile" />
        <property name="cust_image" column="cust_image" />

        <!--配置多对一关系 -->
        <many-to-one name="baseDictSource" class="com.admiral.crm.domain.BaseDict" column="cust_source" />
        <many-to-one name="baseDictIndustry" class="com.admiral.crm.domain.BaseDict" column="cust_industry" />
        <many-to-one name="baseDictLevel" class="com.admiral.crm.domain.BaseDict" column="cust_level" />

    </class>
</hibernate-mapping>
```

- 修改文件上传的方法

```
            //创建目录
            String url = path + realPath;
            File file = new File(url);
            if (!file.exists()) {
                //创建多级目录
                file.mkdirs();
            }

            //进行文件上传
            File destFile = new File(url + "/" + uuidFileName);
            FileUtils.copyFile(upload, destFile);

            //设置文件上传的路径
            customer.setCust_image(url + "/" + uuidFileName);
        }

        customerService.save(customer);
        return NONE;
    }
```

- 配置跳转

Customer.java

```
package com.admiral.crm.web.action;

import java.io.File;
import java.io.IOException;

import org.apache.commons.io.FileUtils;
import org.hibernate.criterion.DetachedCriteria;

import com.admiral.crm.domain.Customer;
import com.admiral.crm.domain.PageBean;
import com.admiral.crm.service.CustomerService;
import com.admiral.crm.utils.UploadUtils;
import com.opensymphony.xwork2.ActionContext;
import com.opensymphony.xwork2.ActionSupport;
import com.opensymphony.xwork2.ModelDriven;

public class CustomerAction extends ActionSupport implements
ModelDriven<Customer> {

    // 模型驱动需要用到的对象
    private Customer customer = new Customer();

    @Override
    public Customer getModel() {
        return customer;
    }

    // 注入 Service
    private CustomerService customerService;

    public void setCustomerService(CustomerService customerService) {
        this.customerService = customerService;
```

```
    }

    // 注入当前页
    private Integer currPage = 1;

    public void setCurrPage(Integer currPage) {
        if (currPage == null) {
            currPage = 1;
        }
        this.currPage = currPage;
    }

    // 注入每页记录数
    private Integer pageSize = 3;

    public void setPageSize(Integer pageSize) {
        if (pageSize == null) {
            pageSize = 1;
        }
        this.pageSize = pageSize;
    }


    /**
     *
     * Title: findAll
     * Description: 客户管理:分页查询
     * @return
     */
    public String findAll() {
        // 接收分页参数
        //
        DetachedCriteria detachedCriteria =
DetachedCriteria.forClass(Customer.class);
        System.out.println(pageSize);
        //调用业务层查询
        PageBean<Customer> pageBean =
customerService.findByPage(detachedCriteria,currPage,pageSize);

        //将分页对象保存到值栈
        ActionContext.getContext().getValueStack().push(pageBean);
        return "findAll";
    }


    /**
     *
     * Title: saveUI
     * Description: 客户管理:跳转到添加页面:saveUI
     * @return
     */
    public String saveUI() {
        return "saveUI";
    }


    /**
     * 文件上传提供的三个属性
```

```
     */
    private String uploadFileName;       // 文件名称
    private File upload;                 // 上传的文件
    private String uploadContextType;    // 文件类型

    public void setUploadFileName(String uploadFileName) {
        this.uploadFileName = uploadFileName;
    }

    public void setUpload(File upload) {
        this.upload = upload;
    }

    public void setUploadContextType(String uploadContextType) {
        this.uploadContextType = uploadContextType;
    }


    /**
     *
     * Title: save
     * Description: 客户管理:保存客户
     * @return
     * @throws IOException
     */
    public String save() throws IOException {
        //文件上传

        if(upload!=null) {
            //上传资质图片
            //设置文件上传的路径
            String path = "d:/upload";

            // 一个目录下存放的相同文件名:文件名随机
            String uuidFileName = UploadUtils.getUuidFileName(uploadFileName);
            // 一个目录下存放的文件过多:目录分离
            String realPath = UploadUtils.getPath(uuidFileName);

            //创建目录
            String url = path + realPath;
            File file = new File(url);
            if (!file.exists()) {
                //创建多级目录
                file.mkdirs();
            }

            //进行文件上传
            File destFile = new File(url + "/" + uuidFileName);
            FileUtils.copyFile(upload, destFile);

            //设置文件上传的路径
            customer.setCust_image(url + "/" + uuidFileName);
        }

        customerService.save(customer);
        return "saveSuccess";
    }
```

```
}
```

struts.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE struts PUBLIC
    "-//Apache Software Foundation//DTD Struts Configuration 2.3//EN"
    "http://struts.apache.org/dtds/struts-2.3.dtd">

<struts>
    <package name="crm" extends="struts-default" namespace="/">

        <action name="user_*" class="userAction" method="{1}">
            <result name="login">/login.jsp</result>
            <result name="success" type="redirect">/index.jsp</result>
        </action>

        <!-- 配置客户模块 -->
        <action name="customer_*" class="customerAction" method="{1}">
            <result name="saveUI">/jsp/customer/add.jsp</result>
            <result name="findAll">/jsp/customer/list.jsp</result>
            <result name="saveSuccess"
type="redirectAction">customer_findAll.action</result>
        </action>

        <!-- 配置字典 -->
        <action name="baseDict_*" class="baseDictAction" method="{1}">

        </action>
    </package>
</struts>
```

1.1.2.4 第四步:设置文件上传拦截器

- 在 struts.xml 中设置文件上传的总大小

```
<struts>

    <constant name="struts.action.extension" value="action" />
    <!-- 设置 Struts2 文件上传总大小 5M -->
    <constant name="struts.multipart.maxSize" value="5242880" />


    <package name="crm" extends="struts-default" namespace="/">

        <action name="user_*" class="userAction" method="{1}">
            <result name="login">/login.jsp</result>
            <result name="success" type="redirect">/index.jsp</result>
        </action>
```

- 在 Customer 配置中引入拦截器栈

```
<!-- 配置客户模块 -->
<action name="customer_*" class="customerAction" method="{1}">
    <result name="saveUI">/jsp/customer/add.jsp</result>
    <result name="findAll">/jsp/customer/list.jsp</result>
    <result name="saveSuccess" type="redirectAction">customer_findAll.action</result>

    <interceptor-ref name="defaultStack">
        <param name="fileUpload.maximumSize">2097152</param>
        <param name="fileUpload.allowedExtensions">.jpg,.png</param>
    </interceptor-ref>
</action>

<!-- 配置字典 -->
```

- 配置 input 视图

```
<!-- 配置客户模块 -->
<action name="customer_*" class="customerAction" method="{1}">
    <result name="saveUI">/jsp/customer/add.jsp</result>
    <result name="findAll">/jsp/customer/list.jsp</result>
    <result name="saveSuccess" type="redirectAction">customer_findAll.action</result>
    <result name="input">/jsp/customer/add.jsp</result>

    <interceptor-ref name="defaultStack">
        <param name="fileUpload.maximumSize">2097152</param>
        <param name="fileUpload.allowedExtensions">.jpg,.png</param>
    </interceptor-ref>
</action>
```

- 在页面显示错误信息

```
6
7       <s:fielderror />
```

UploadUtils.java

```
package com.admiral.crm.utils;

import java.util.UUID;

/**
 *
 * @Description: 文件上传的工具类
 * @author Admiral
 */
public class UploadUtils {

    /**
     *
     * Title: getUuidFileName
     * Description: 解决目录下文件名重复的问题
```

```
     * @param fileName :文件名
     * @return : 随机产生的唯一文件名
     */
    public static String getUuidFileName(String fileName) {
        int indexOf = fileName.lastIndexOf(".");
        String extions = fileName.substring(indexOf);
        return UUID.randomUUID().toString().replace("-", "") + extions;

    }

    public static String getPath(String uuidFileName) {
        int code1 = uuidFileName.hashCode();
        int d1 = code1 & 0xf;        //作为1级目录
        int code2 = code1 >>> 4;
        int d2 = code2 & 0xf;        //作为2级目录
        return "/" + d1 + "/" + d2;
    }

    public static void main(String[] args) {
        System.out.println(UploadUtils.getUuidFileName("aa.txt"));
    }
}
```

1.2 客户管理 - 删除客户

1.2.1 客户删除操作

1.2.1.1 修改列表页面上链接地址

```
<a href="${pageContext.request.contextPath }/customer_delete.action?cust_id=<s:property value="cust_id"/>">删除</a>
```

1.2.1.2 编写 Action 中的 delete 方法

```java
/**
 *
 * Title: delete
 * Description: 删除客户的方法
 * @return
 */
public String delete() {
    //先查询再删除
    customer = customerService.findById(customer.getCust_id());
    //删除图片
    if(customer.getCust_image() != null) {
        File file = new File(customer.getCust_image());
        if (file != null) {
            file.delete();
        }
    }

    //删除客户
    customerService.delete(customer);

    return "deleteSuccess";
}
```

1.2.1.3 编写 Service

```java
@Override
public Customer findById(Long cust_id) {
    return customerDao.findById(cust_id);
}

@Override
public void delete(Customer customer) {
    customerDao.delete(customer);
}
```

1.2.1.4 编写 Dao

```java
@Override
public Customer findById(Long cust_id) {
    return this.getHibernateTemplate().get(Customer.class, cust_id);
}

@Override
public void delete(Customer customer) {
    this.getHibernateTemplate().delete(customer);
}
```

1.2.1.5 配置跳转

```
<!-- 配置客户模块 -->
<action name="customer_*" class="customerAction" method="{1}">
    <result name="saveUI">/jsp/customer/add.jsp</result>
    <result name="findAll">/jsp/customer/list.jsp</result>
    <result name="saveSuccess" type="redirectAction">customer_findAll.action</result>
    <result name="input">/jsp/customer/add.jsp</result>
    <result name="deleteSuccess" type="redirectAction">customer_findAll.action</result>

    <interceptor-ref name="defaultStack">
        <param name="fileUpload.maximumSize">2097152</param>
        <param name="fileUpload.allowedExtensions">.jpg,.png</param>
    </interceptor-ref>
</action>
```

1.3 客户管理 - 修改客户

1.3.1 修改客户代码实现

1.3.1.1 第一步:修改list页面链接地址

```
<a href="${pageContext.request.contextPath }/customer_edit.action?cust_id=<s:property value="cust_id"/>">修改</a>
  
```

1.3.1.2 第二步:编写 Action 中的 edit 方法

```
/**
 *
 * Title: edit
 * Description:
 * @return
 */
public String edit() {
    // 根据 ID 查询,跳转页面,回显数据
    customer = customerService.findById(customer.getCust_id());
    // 将 customer 传递到页面
    // 两种方式:第一种:手动压栈.  第二种:因为模型驱动的对象,默认就在栈顶
    // 如果使用第一种方式:回显数据: <s:property value="cust_name"/>
    // 如果使用第二种方式:回显数据: <s:property value="model.cust_name"/>

    return "editSuccess";
}
```

- 配置跳转

```
<!-- 配置客户模块 -->
<action name="customer_*" class="customerAction" method="{1}">
    <result name="saveUI">/jsp/customer/add.jsp</result>
    <result name="findAll">/jsp/customer/list.jsp</result>
```

```
            <result name="editSuccess">/jsp/customer/edit.jsp</result>
            <result name="saveSuccess"
type="redirectAction">customer_findAll.action</result>
            <result name="input">/jsp/customer/add.jsp</result>
            <result name="deleteSuccess"
type="redirectAction">customer_findAll.action</result>


            <interceptor-ref name="defaultStack">
                <param name="fileUpload.maximumSize">2097152</param>
                <param name="fileUpload.allowedExtensions">.jpg,.png</param>
            </interceptor-ref>
        </action>
```

1.3.1.3 第三步:在页面回显数据

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib uri="/struts-tags" prefix="s"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<TITLE>添加客户</TITLE>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<LINK href="${pageContext.request.contextPath }/css/Style.css" type=text/css
rel=stylesheet>
<LINK href="${pageContext.request.contextPath }/css/Manage.css" type=text/css
    rel=stylesheet>

<script type="text/javascript" src="${pageContext.request.contextPath
}/js/jquery-1.11.3.min.js"></script>
<script type="text/javascript">
    $(function(){
        // 页面加载函数就会执行:
        // 页面加载，异步查询字典数据:
        // 加载客户来源
        $.post("${pageContext.request.contextPath
}/baseDict_findByTypeCode.action",{"dict_type_code":"002"},function(data){
            // 遍历json的数据:
            $(data).each(function(i,n){
                $("#cust_source").append("<option
value='"+n.dict_id+"'>"+n.dict_item_name+"</option>");
            });

            //使用 EL 获取值栈的数据
            $("#cust_source
option[value='${model.baseDictSource.dict_id}']").prop("selected","selected");
```

```
        },"json");
        $.post("${pageContext.request.contextPath
}/baseDict_findByTypeCode.action",{"dict_type_code":"006"},function(data){
            // 遍历json的数据:
            $(data).each(function(i,n){
                $("#cust_level").append("<option
value='"+n.dict_id+"'>"+n.dict_item_name+"</option>");
            });

            //使用 EL 获取值栈的数据
            $("#cust_level
option[value='${model.baseDictLevel.dict_id}']").prop("selected","selected");
        },"json");
        $.post("${pageContext.request.contextPath
}/baseDict_findByTypeCode.action",{"dict_type_code":"001"},function(data){
            // 遍历json的数据:
            $(data).each(function(i,n){
                $("#cust_industry").append("<option
value='"+n.dict_id+"'>"+n.dict_item_name+"</option>");
            });

            //使用 EL 获取值栈的数据
            $("#cust_industry
option[value='${model.baseDictIndustry.dict_id}']").prop("selected","selected");
        },"json");
    });
</script>

<META content="MSHTML 6.00.2900.3492" name=GENERATOR>
</HEAD>
<BODY>
    <s:form id="form1" name="form1" action=""
        method="post" enctype="multipart/form-data" theme="simple">

        <s:hidden name="cust_id" value="%{model.cust_id}"/>
        <s:hidden name="cust_image" value="%{model.cust_image}"/>

        <s:fielderror />
        <TABLE cellSpacing=0 cellPadding=0 width="98%" border=0>
            <TBODY>
                <TR>
                    <TD width=15><IMG src="${pageContext.request.contextPath
}/images/new_019.jpg"
                        border=0></TD>
                    <TD width="100%"
background="${pageContext.request.contextPath }/images/new_020.jpg"
                        height=20></TD>
                    <TD width=15><IMG src="${pageContext.request.contextPath
}/images/new_021.jpg"
                        border=0></TD>
                </TR>
            </TBODY>
        </TABLE>
        <TABLE cellSpacing=0 cellPadding=0 width="98%" border=0>
            <TBODY>
                <TR>
                    <TD width=15 background=${pageContext.request.contextPath
}/images/new_022.jpg><IMG
```

```
                        src="${pageContext.request.contextPath
}/images/new_022.jpg" border=0></TD>
                    <TD vAlign=top width="100%" bgColor=#ffffff>
                        <TABLE cellSpacing=0 cellPadding=5 width="100%"
border=0>
                            <TR>
                                <TD class=manageHead>当前位置：客户管理 &gt; 添加客户
</TD>
                            </TR>
                            <TR>
                                <TD height=2></TD>
                            </TR>
                        </TABLE>

                        <TABLE cellSpacing=0 cellPadding=5  border=0>


                            <TR>
                                <td>客户名称：</td>
                                <td>
                                    <s:textfield class="textbox" id="sChannel2"
style="WIDTH: 180px" maxlength="50" name="cust_name" value="%{model.cust_name
}"/>
                                </td>
                                <td>客户级别 ：</td>
                                <td>
                                    <select id="cust_level"
name="baseDictLevel.dict_id">
                                        <option value="">-请选择-</option>
                                    </select>
                                </td>
                            </TR>

                            <TR>

                                <td>信息来源 ：</td>
                                <td>
                                <select id="cust_source"
name="baseDictSource.dict_id">
                                    <option value="">-请选择-</option>
                                </select>
                                </td>
                                <td>所属行业 ：</td>
                                <td>
                                    <select id="cust_industry"
name="baseDictIndustry.dict_id">
                                        <option value="">-请选择-</option>
                                    </select>
                                </td>
                            </TR>

                            <TR>


                                <td>固定电话 ：</td>
                                <td>
                                    <s:textfield class="textbox" style="WIDTH:
180px" maxlength="50" name="cust_phone" value="%{model.cust_phone}"/>
```

```
                                </td>
                                <td>移动电话 ： </td>
                                <td>
                                    <s:textfield class="textbox" style="WIDTH:
180px" maxlength="50" name="cust_mobile" value="%{model.cust_mobile}"/>
                                </td>
                            </TR>

                            <TR>


                                <td>客户资质 ： </td>
                                <td colspan="3">
                                    <input type="file" name="upload" />
                                </td>
                            </TR>


                            <tr>
                                <td rowspan=2>
                                <INPUT class=button id=sButton2 type=submit
                                                    value=" 保存 "
name=sButton2>
                                </td>
                            </tr>
                        </TABLE>


                    </TD>
                    <TD width=15 background="${pageContext.request.contextPath
}/images/new_023.jpg">
                    <IMG src="${pageContext.request.contextPath
}/images/new_023.jpg" border=0></TD>
                </TR>
            </TBODY>
        </TABLE>
        <TABLE cellSpacing=0 cellPadding=0 width="98%" border=0>
            <TBODY>
                <TR>
                    <TD width=15><IMG src="${pageContext.request.contextPath
}/images/new_024.jpg"
                        border=0></TD>
                    <TD align=middle width="100%"
                        background="${pageContext.request.contextPath
}/images/new_025.jpg" height=15></TD>
                    <TD width=15><IMG src="${pageContext.request.contextPath
}/images/new_026.jpg"
                        border=0></TD>
                </TR>
            </TBODY>
        </TABLE>
    </s:form>
</BODY>
</HTML>
```

1.3.1.4 第四步:修改编辑页面的提交路径

```
<BODY>
    <s:form id="form1" name="form1" action="customer_update" namespace="/"
        method="post" enctype="multipart/form-data" theme="simple">
```

1.3.1.5 编写 Action 中的 update 方法

```
    /**
     *
     * Title: update
     * Description: 修改客户的方法
     * @return
     * @throws IOException
     */
    public String update() throws IOException {
        //判断文件项是否已经选择,如果选择了,删除原有文件,上传新文件,如果没有选择,使用原有文件
即可

        if(upload!=null) {
            //已经选择了

            //删除原有文件
            String cust_image = customer.getCust_image();//从隐藏域中获取原有路径
            if(cust_image!=null || "".equals(cust_image)) {
                File file = new File(cust_image);
                if (file != null) {
                    file.delete();
                }
            }
            //文件上传:
            //设置文件上传的路径
            String path = "d:/upload";

            // 一个目录下存放的相同文件名:文件名随机
            String uuidFileName = UploadUtils.getUuidFileName(uploadFileName);
            // 一个目录下存放的文件过多:目录分离
            String realPath = UploadUtils.getPath(uuidFileName);

            //创建目录
            String url = path + realPath;
            File file = new File(url);
            if (!file.exists()) {
                //创建多级目录
                file.mkdirs();
            }

            //进行文件上传
            File destFile = new File(url + "/" + uuidFileName);
            FileUtils.copyFile(upload, destFile);

            //设置文件上传的路径
            customer.setCust_image(url + "/" + uuidFileName);
        }
        //修改客户
        customerService.update(customer);
```

```
        return "updateSuccess";
    }
```

- 配置跳转

```
        <!-- 配置客户模块 -->
        <action name="customer_*" class="customerAction" method="{1}">
            <result name="saveUI">/jsp/customer/add.jsp</result>
            <result name="findAll">/jsp/customer/list.jsp</result>
            <result name="editSuccess">/jsp/customer/edit.jsp</result>
            <result name="saveSuccess"
type="redirectAction">customer_findAll.action</result>
            <result name="updateSuccess"
type="redirectAction">customer_findAll.action</result>
            <result name="input">/jsp/customer/add.jsp</result>
            <result name="deleteSuccess"
type="redirectAction">customer_findAll.action</result>


            <interceptor-ref name="defaultStack">
                <param name="fileUpload.maximumSize">2097152</param>
                <param name="fileUpload.allowedExtensions">.jpg,.png</param>
            </interceptor-ref>
        </action>
```

1.4 客户管理 - 条件查询客户

1.4.1 客户管理:条件查询

1.4.1.1 在列表页面准备条件

```
<TR>
    <TD>客户名称：</TD>
    <TD><INPUT class=textbox id=sChannel2
        style="WIDTH: 80px" maxLength=50 name="custName"></TD>
    <TD>客户来源：</TD>
    <TD>
        <select id="cust_source" name="baseDictSource.dict_id">
            <option value="">-请选择-</option>
        </select>
    </TD>
    <TD>客户级别：</TD>
    <TD>
        <select id="cust_level" name="baseDictLevel.dict_id">
            <option value="">-请选择-</option>
        </select>
    </TD>
    <TD>所属行业：</TD>
    <TD>
        <select id="cust_industry" name="baseDictIndustry.dict_id">
            <option value="">-请选择-</option>
        </select>
    </TD>

    <TD><INPUT class=button id=sButton2 type=submit
        value=" 筛选 " name=sButton2></TD>
</TR>
```

1.4.1.2 异步加载数据

```
<script type="text/javascript" src="${pageContext.request.contextPath
}/js/jquery-1.11.3.min.js"></script>
<script type="text/javascript">
    $(function(){
        // 页面加载函数就会执行：
        // 页面加载，异步查询字典数据：
        // 加载客户来源
        $.post("${pageContext.request.contextPath
}/baseDict_findByTypeCode.action",{"dict_type_code":"002"},function(data){
            // 遍历json的数据：
            $(data).each(function(i,n){
                $("#cust_source").append("<option
value='"+n.dict_id+"'>"+n.dict_item_name+"</option>");
            });
        },"json");
        $.post("${pageContext.request.contextPath
}/baseDict_findByTypeCode.action",{"dict_type_code":"006"},function(data){
            // 遍历json的数据：
            $(data).each(function(i,n){
                $("#cust_level").append("<option
value='"+n.dict_id+"'>"+n.dict_item_name+"</option>");
            });
        },"json");
        $.post("${pageContext.request.contextPath
}/baseDict_findByTypeCode.action",{"dict_type_code":"001"},function(data){
```

```
            // 遍历json的数据:
            $(data).each(function(i,n){
                $("#cust_industry").append("<option
value='"+n.dict_id+"'>"+n.dict_item_name+"</option>");
            });
        },"json");
    });
</script>
```

1.4.1.3 改写 Action 中的 findAll 方法

```
    /**
     *
     * Title: findAll
     * Description: 客户管理:分页查询
     * @return
     */
    public String findAll() {
        // 接收分页参数
        // (条件查询,带分页)
        DetachedCriteria detachedCriteria =
DetachedCriteria.forClass(Customer.class);
        // 在web层设置条件
        if(customer.getCust_name() != null) {
            detachedCriteria.add(Restrictions.like("cust_name", "%"+
customer.getCust_name() +"%"));
        }

        if(customer.getBaseDictSource()!=null) {
            if(customer.getBaseDictSource().getDict_id()!=null &&
!"".equals(customer.getBaseDictSource().getDict_id())) {
                detachedCriteria.add(Restrictions.eq("baseDictSource.dict_id",
customer.getBaseDictSource().getDict_id()));
            }
        }

        if(customer.getBaseDictLevel()!=null) {
            if(customer.getBaseDictLevel().getDict_id()!=null &&
!"".equals(customer.getBaseDictLevel().getDict_id())) {
                detachedCriteria.add(Restrictions.eq("baseDictLevel.dict_id",
customer.getBaseDictLevel().getDict_id()));
            }
        }

        if(customer.getBaseDictIndustry()!=null) {
            if(customer.getBaseDictIndustry().getDict_id()!=null &&
!"".equals(customer.getBaseDictIndustry().getDict_id())) {
```

```
                detachedCriteria.add(Restrictions.eq("baseDictIndustry.dict_id",
customer.getBaseDictIndustry().getDict_id()));
            }
        }

        System.out.println(pageSize);
        //调用业务层查询
        PageBean<Customer> pageBean =
customerService.findByPage(detachedCriteria,currPage,pageSize);

        //将分页对象保存到值栈
        ActionContext.getContext().getValueStack().push(pageBean);
        return "findAll";
    }
```

1.3.1.6 在条件上回显数据

```
<TD>客户名称：</TD>
<TD><INPUT class=textbox id=sChannel2
    style="WIDTH: 80px" maxLength=50 name="cust_name" value='<s:property value="model.cust_name"/>'></TD>
<TD>客户来源：</TD>
```

```
<script type="text/javascript">
    $(function(){
        // 页面加载函数就会执行：
        // 页面加载，异步查询字典数据：
        // 加载客户来源
        $.post("${pageContext.request.contextPath
}/baseDict_findByTypeCode.action",{"dict_type_code":"002"},function(data){
            // 遍历json的数据：
            $(data).each(function(i,n){
                $("#cust_source").append("<option
value='"+n.dict_id+"'>"+n.dict_item_name+"</option>");
            });

            //回显数据
            $("#cust_source
option[value='${model.baseDictSource.dict_id}']").prop("selected","selected");
        },"json");
        $.post("${pageContext.request.contextPath
}/baseDict_findByTypeCode.action",{"dict_type_code":"006"},function(data){
            // 遍历json的数据：
            $(data).each(function(i,n){
                $("#cust_level").append("<option
value='"+n.dict_id+"'>"+n.dict_item_name+"</option>");
            });

            //回显数据
            $("#cust_level
option[value='${model.baseDictLevel.dict_id}']").prop("selected","selected");
        },"json");
```

```
        $.post("${pageContext.request.contextPath
}/baseDict_findByTypeCode.action",{"dict_type_code":"001"},function(data){
            // 遍历json的数据:
            $(data).each(function(i,n){
                $("#cust_industry").append("<option
value='"+n.dict_id+"'>"+n.dict_item_name+"</option>");
            });

            //回显数据
            $("#cust_industry
option[value='${model.baseDictIndustry.dict_id}']").prop("selected","selected");
        },"json");
    });
</script>
```

1.5 联系人管理 - 查询列表

1.5.1 准备工作

1.5.1.1 创建表

```
CREATE TABLE `cst_linkman` (
  `lkm_id` bigint(32) NOT NULL AUTO_INCREMENT COMMENT '联系人编号(主键)',
  `lkm_name` varchar(16) DEFAULT NULL COMMENT '联系人姓名',
  `lkm_cust_id` bigint(32) NOT NULL COMMENT '客户id',
  `lkm_gender` char(1) DEFAULT NULL COMMENT '联系人性别',
  `lkm_phone` varchar(16) DEFAULT NULL COMMENT '联系人办公电话',
  `lkm_mobile` varchar(16) DEFAULT NULL COMMENT '联系人手机',
  `lkm_email` varchar(64) DEFAULT NULL COMMENT '联系人邮箱',
  `lkm_qq` varchar(16) DEFAULT NULL COMMENT '联系人qq',
  `lkm_position` varchar(16) DEFAULT NULL COMMENT '联系人职位',
  `lkm_memo` varchar(512) DEFAULT NULL COMMENT '联系人备注',
  PRIMARY KEY (`lkm_id`),
  KEY `FK_cst_linkman_lkm_cust_id` (`lkm_cust_id`),
  CONSTRAINT `FK_cst_linkman_lkm_cust_id` FOREIGN KEY (`lkm_cust_id`) REFERENCES
`cst_customer` (`cust_id`) ON DELETE NO ACTION ON UPDATE NO ACTION
) ENGINE=InnoDB AUTO_INCREMENT=3 DEFAULT CHARSET=utf8;
```

1.5.1.2 创建实体和映射

- 创建实体

```
package com.admiral.crm.domain;

/**
 *
 * @Description: 联系人实体
 * @author Admiral
 *
 * CREATE TABLE `cst_linkman` (
  `lkm_id` bigint(32) NOT NULL AUTO_INCREMENT COMMENT '联系人编号(主键)',
  `lkm_name` varchar(16) DEFAULT NULL COMMENT '联系人姓名',
  `lkm_cust_id` bigint(32) NOT NULL COMMENT '客户id',
  `lkm_gender` char(1) DEFAULT NULL COMMENT '联系人性别',
  `lkm_phone` varchar(16) DEFAULT NULL COMMENT '联系人办公电话',
  `lkm_mobile` varchar(16) DEFAULT NULL COMMENT '联系人手机',
  `lkm_email` varchar(64) DEFAULT NULL COMMENT '联系人邮箱',
  `lkm_qq` varchar(16) DEFAULT NULL COMMENT '联系人qq',
  `lkm_position` varchar(16) DEFAULT NULL COMMENT '联系人职位',
  `lkm_memo` varchar(512) DEFAULT NULL COMMENT '联系人备注',
  PRIMARY KEY (`lkm_id`),
  KEY `FK_cst_linkman_lkm_cust_id` (`lkm_cust_id`),
  CONSTRAINT `FK_cst_linkman_lkm_cust_id` FOREIGN KEY (`lkm_cust_id`) REFERENCES
`cst_customer` (`cust_id`) ON DELETE NO ACTION ON UPDATE NO ACTION
) ENGINE=InnoDB AUTO_INCREMENT=3 DEFAULT CHARSET=utf8;
 */
public class LinkMan {
    private Long lkm_id;
    private String lkm_name;
    private String lkm_gender;
    private String lkm_phone;
    private String lkm_mobile;
    private String lkm_email;
    private String lkm_qq;
    private String lkm_position;
    private String lkm_memo;

    //方式客户的对象
    private Customer customer;

    public Long getLkm_id() {
        return lkm_id;
    }

    public void setLkm_id(Long lkm_id) {
        this.lkm_id = lkm_id;
    }

    public String getLkm_name() {
        return lkm_name;
    }

    public void setLkm_name(String lkm_name) {
        this.lkm_name = lkm_name;
    }
```

```
    public String getLkm_gender() {
        return lkm_gender;
    }

    public void setLkm_gender(String lkm_gender) {
        this.lkm_gender = lkm_gender;
    }

    public String getLkm_phone() {
        return lkm_phone;
    }

    public void setLkm_phone(String lkm_phone) {
        this.lkm_phone = lkm_phone;
    }

    public String getLkm_mobile() {
        return lkm_mobile;
    }

    public void setLkm_mobile(String lkm_mobile) {
        this.lkm_mobile = lkm_mobile;
    }

    public String getLkm_email() {
        return lkm_email;
    }

    public void setLkm_email(String lkm_email) {
        this.lkm_email = lkm_email;
    }

    public String getLkm_qq() {
        return lkm_qq;
    }

    public void setLkm_qq(String lkm_qq) {
        this.lkm_qq = lkm_qq;
    }

    public String getLkm_position() {
        return lkm_position;
    }

    public void setLkm_position(String lkm_position) {
        this.lkm_position = lkm_position;
    }

    public String getLkm_memo() {
        return lkm_memo;
    }

    public void setLkm_memo(String lkm_memo) {
        this.lkm_memo = lkm_memo;
    }

    public Customer getCustomer() {
```

```
        return customer;
    }

    public void setCustomer(Customer customer) {
        this.customer = customer;
    }
}
```

- 创建映射

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-mapping PUBLIC
    "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
    "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">

<hibernate-mapping>
    <class name="com.admiral.crm.domain.LinkMan" table="cst_linkman">
        <id name="lkm_id" column="lkm_id">
            <generator class="native" />
        </id>

        <property name="lkm_name" column="lkm_name" />
        <property name="lkm_gender" column="lkm_gender" />
        <property name="lkm_phone" column="lkm_phone" />
        <property name="lkm_mobile" column="lkm_mobile" />
        <property name="lkm_email" column="lkm_email" />
        <property name="lkm_qq" column="lkm_qq" />
        <property name="lkm_position" column="lkm_position" />
        <property name="lkm_memo" column="lkm_memo" />

        <many-to-one name="customer" class="com.admiral.crm.domain.Customer"
column="lkm_cust_id" />
    </class>
</hibernate-mapping>
```

- 修改客户的实体

```

package com.admiral.crm.domain;

import java.util.HashSet;
import java.util.Set;
```

```java
public class Customer {

    private Long cust_id;
    private String cust_name;
//  private String cust_source;
//  private String cust_industry;
//  private String cust_level;
    private String cust_phone;
    private String cust_mobile;
    private String cust_image;//文件上传的路径

    /*
     * 客户和字典是多对一关系,需要在多的一方放置一的一方的对象
     */
    private BaseDict baseDictSource;
    private BaseDict baseDictIndustry;
    private BaseDict baseDictLevel;

    /*
     * 放置联系人的集合
     */
    private Set<LinkMan> linkMans = new HashSet<LinkMan>();

    public BaseDict getBaseDictSource() {
        return baseDictSource;
    }

    public void setBaseDictSource(BaseDict baseDictSource) {
        this.baseDictSource = baseDictSource;
    }

    public BaseDict getBaseDictIndustry() {
        return baseDictIndustry;
    }

    public void setBaseDictIndustry(BaseDict baseDictIndustry) {
        this.baseDictIndustry = baseDictIndustry;
    }

    public BaseDict getBaseDictLevel() {
        return baseDictLevel;
    }

    public void setBaseDictLevel(BaseDict baseDictLevel) {
        this.baseDictLevel = baseDictLevel;
    }

    public Long getCust_id() {
        return cust_id;
    }

    public void setCust_id(Long cust_id) {
        this.cust_id = cust_id;
    }

    public String getCust_name() {
        return cust_name;
    }
```

```
    public void setCust_name(String cust_name) {
        this.cust_name = cust_name;
    }

    public String getCust_phone() {
        return cust_phone;
    }

    public void setCust_phone(String cust_phone) {
        this.cust_phone = cust_phone;
    }

    public String getCust_mobile() {
        return cust_mobile;
    }

    public void setCust_mobile(String cust_mobile) {
        this.cust_mobile = cust_mobile;
    }

    public String getCust_image() {
        return cust_image;
    }

    public void setCust_image(String cust_image) {
        this.cust_image = cust_image;
    }

    public Set<LinkMan> getLinkMans() {
        return linkMans;
    }

    public void setLinkMans(Set<LinkMan> linkMans) {
        this.linkMans = linkMans;
    }
}
```

- 修改客户的映射文件

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-mapping PUBLIC
    "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
    "http://www.hibernate.org/dtd/hibernate-mapping-3.0.dtd">

<hibernate-mapping>
    <class name="com.admiral.crm.domain.Customer" table="cst_customer">
        <!--建立类中的属性与表中的主键的映射关系 -->
```

```
        <id name="cust_id" column="cust_id">
            <!--主键的生成策略 -->
            <generator class="native"></generator>
        </id>

        <!-- 建立类中的普通属性与表中的字段的映射 -->
        <property name="cust_name" column="cust_name" />
        <!-- <property name="cust_source" column="cust_source" />
        <property name="cust_industry" column="cust_industry" />
        <property name="cust_level" column="cust_level" /> -->
        <property name="cust_phone" column="cust_phone" />
        <property name="cust_mobile" column="cust_mobile" />
        <property name="cust_image" column="cust_image" />

        <!-- 配置多对一关系 -->
        <many-to-one name="baseDictSource"
class="com.admiral.crm.domain.BaseDict" column="cust_source" />
        <many-to-one name="baseDictIndustry"
class="com.admiral.crm.domain.BaseDict" column="cust_industry" />
        <many-to-one name="baseDictLevel"
class="com.admiral.crm.domain.BaseDict" column="cust_level" />

        <!-- 配置与联系人的关系映射 -->
        <set name="linkMans">
            <key column="lkm_cust_id"/>
            <one-to-many class="com.admiral.crm.domain.LinkMan"/>
        </set>

    </class>
</hibernate-mapping>
```

- 在 Spring 中加入映射文件

```
        <property name="mappingResources">
            <list>
                <value>com/admiral/crm/domain/User.hbm.xml</value>
                <value>com/admiral/crm/domain/Customer.hbm.xml</value>
                <value>com/admiral/crm/domain/BaseDict.hbm.xml</value>
                <value>com/admiral/crm/domain/LinkMan.hbm.xml</value>
            </list>
        </property>
    </bean>
```

1.5.1.3 创建相关类

- 联系人Dao的接口

```java
package com.admiral.crm.dao;

/**
 *
 * @Description: 联系人Dao的接口
 * @author Admiral
 */
public interface LinkManDao {

}
```

- 联系人Dao的实现类

```java
package com.admiral.crm.dao.impl;

import org.springframework.orm.hibernate5.support.HibernateDaoSupport;

import com.admiral.crm.dao.LinkManDao;

/**
 *
 * @Description: 联系人Dao的实现类
 * @author Admiral
 */
public class LinkManDaoImpl extends HibernateDaoSupport implements LinkManDao {

}
```

- 联系人Service的接口

```java
package com.admiral.crm.service;

/**
 *
 * @Description: 联系人Service层的接口
 * @author Admiral
 */
public interface LinkManService {

}
```

- 联系人Service的实现类

```java
package com.admiral.crm.service.impl;

import com.admiral.crm.dao.LinkManDao;
import com.admiral.crm.service.LinkManService;
```

```java
/**
 *
 * @Description: 联系人Service层的实现类
 * @author Admiral
 */
public class LinkManServiceImpl implements LinkManService {

    //注入 Dao
    private LinkManDao linkManDao;

    public void setLinkManDao(LinkManDao linkManDao) {
        this.linkManDao = linkManDao;
    }

}
```

- 联系人的Action

```java
package com.admiral.crm.web.action;

import com.admiral.crm.domain.LinkMan;
import com.admiral.crm.service.LinkManService;
import com.opensymphony.xwork2.ActionSupport;
import com.opensymphony.xwork2.ModelDriven;

public class LinkManAction extends ActionSupport implements ModelDriven<LinkMan>
{

    //模型驱动使用的对象
    private LinkMan linkMan = new LinkMan();
    @Override
    public LinkMan getModel() {
        return linkMan;
    }

    //注入 Service
    private LinkManService linkManService;

    public void setLinkManService(LinkManService linkManService) {
        this.linkManService = linkManService;
    }


}
```

1.5.1.4 将相关类交给 Spring 管理

```
    <!-- ================  LinkMan 模块的相关配置  ====================== -->
    <bean id="linkManAction" class="com.admiral.crm.web.action.LinkManAction">
        <property name="linkManService" ref="linkManService" />
    </bean>
    <bean id="linkManService"
class="com.admiral.crm.service.impl.LinkManServiceImpl">
        <property name="linkManDao" ref="linkManDao" />
    </bean>
    <bean id="linkManDao" class="com.admiral.crm.dao.impl.LinkManDaoImpl">
        <property name="sessionFactory" ref="sessionFactory" />
    </bean>
```

- struts.xml

```
        <!-- 配置联系人模块 -->
        <action name="linkMan_*" class="linkManAction" method="{1}">

        </action>
```

1.5.2 查询联系人的列表

1.5.2.1 修改 menu.jsp 的链接

```
<TR>
    <TD class=menuSmall><A class=style2 href="${pageContext.request.contextPath}/linkMan_findAll.action"
        target=main>－联系人列表</A></TD>
</TR>
```

1.5.2.2 编写 Action 的 findAll 方法

```
package com.admiral.crm.web.action;

import org.hibernate.criterion.DetachedCriteria;

import com.admiral.crm.domain.LinkMan;
import com.admiral.crm.domain.PageBean;
import com.admiral.crm.service.LinkManService;
import com.opensymphony.xwork2.ActionContext;
import com.opensymphony.xwork2.ActionSupport;
import com.opensymphony.xwork2.ModelDriven;

public class LinkManAction extends ActionSupport implements ModelDriven<LinkMan>
{

    //模型驱动使用的对象
    private LinkMan linkMan = new LinkMan();
```

```
    @Override
    public LinkMan getModel() {
        return linkMan;
    }

    //注入 Service
    private LinkManService linkManService;

    public void setLinkManService(LinkManService linkManService) {
        this.linkManService = linkManService;
    }

    private Integer currPage = 1;
    private Integer pageSize = 3;

    public void setCurrPage(Integer currPage) {
        if(currPage==null) {
            currPage = 1;
        }
        this.currPage = currPage;
    }

    public void setPageSize(Integer pageSize) {
        if (pageSize == null) {
            pageSize = 3;
        }
        this.pageSize = pageSize;
    }

    /**
     *
     * Title: findAll
     * Description: 待条件分页查询联系人
     * @return
     */
    public String findAll() {
        //创建离线条件查询
        DetachedCriteria detachedCriteria =
DetachedCriteria.forClass(LinkMan.class);
        //设置条件

        //调用业务层
        PageBean<LinkMan> pageBean =
linkManService.findAll(detachedCriteria,currPage,pageSize);
        //将 pageBean 压入值栈中
        ActionContext.getContext().getValueStack().push(pageBean);
        return "findAll";
    }

}
```

1.5.2.3 编写 Service

```
package com.admiral.crm.service.impl;

import java.util.List;

import org.hibernate.criterion.DetachedCriteria;

import com.admiral.crm.dao.LinkManDao;
import com.admiral.crm.domain.LinkMan;
import com.admiral.crm.domain.PageBean;
import com.admiral.crm.service.LinkManService;

/**
 *
 * @Description: 联系人Service层的实现类
 * @author Admiral
 */
public class LinkManServiceImpl implements LinkManService {

    //注入 Dao
    private LinkManDao linkManDao;

    public void setLinkManDao(LinkManDao linkManDao) {
        this.linkManDao = linkManDao;
    }

    @Override
    public PageBean<LinkMan> findAll(DetachedCriteria detachedCriteria, Integer
currPage, Integer pageSize) {
        PageBean<LinkMan> pageBean = new PageBean<LinkMan>();
        //设置当前页
        pageBean.setCurrPage(currPage);
        //设置每页的记录数
        pageBean.setPageSize(pageSize);

        //设置总记录数
        Integer totalCount = linkManDao.findCount(detachedCriteria);
        pageBean.setTotalCount(totalCount);

        //设置总页数
        Double num = Math.ceil(totalCount.doubleValue()/pageSize);
        pageBean.setTotalPage(num.intValue());

        //设置每页显示的数据
        Integer begin = (currPage-1) * pageSize;
        List<LinkMan> linkMans =
linkManDao.findByPage(detachedCriteria,begin,pageSize);
        pageBean.setList(linkMans);

        return pageBean;
    }

}
```

1.5.2.4 编写 Dao

```
package com.admiral.crm.dao.impl;

import java.util.List;

import org.hibernate.criterion.DetachedCriteria;
import org.hibernate.criterion.Projections;
import org.springframework.orm.hibernate5.support.HibernateDaoSupport;

import com.admiral.crm.dao.LinkManDao;
import com.admiral.crm.domain.LinkMan;

/**
 *
 * @Description: 联系人Dao的实现类
 * @author Admiral
 */
public class LinkManDaoImpl extends HibernateDaoSupport implements LinkManDao {

    @Override
    public Integer findCount(DetachedCriteria detachedCriteria) {
        detachedCriteria.setProjection(Projections.rowCount());
        List<Long> list = (List<Long>)
this.getHibernateTemplate().findByCriteria(detachedCriteria);
        if(list!=null && list.size()>0) {
            return list.get(0).intValue();
        }
        return null;
    }

    @Override
    public List<LinkMan> findByPage(DetachedCriteria detachedCriteria, Integer
begin, Integer pageSize) {
        detachedCriteria.setProjection(null);
        return (List<LinkMan>)
this.getHibernateTemplate().findByCriteria(detachedCriteria, begin, pageSize);
    }

}
```

1.5.3 查询联系人前台

1.5.3.1 配置 Action 的跳转

```
        <!-- 配置联系人模块 -->
        <action name="linkMan_*" class="linkManAction" method="{1}">
            <result name="findAll">/jsp/linkman/list.jsp</result>
        </action>
```

1.5.3.2 修改 list.jsp 页面

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib uri="/struts-tags" prefix="s"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<TITLE>联系人列表</TITLE>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<LINK href="${pageContext.request.contextPath }/css/Style.css" type=text/css
rel=stylesheet>
<LINK href="${pageContext.request.contextPath }/css/Manage.css" type=text/css
    rel=stylesheet>
<script type="text/javascript" src="${pageContext.request.contextPath
}/js/jquery-1.4.4.min.js"></script>
<SCRIPT language=javascript>
    function to_page(page){
        if(page){
            $("#page").val(page);
        }
        document.customerForm.submit();

    }
</SCRIPT>

<META content="MSHTML 6.00.2900.3492" name=GENERATOR>
</HEAD>
<BODY>
    <FORM id="customerForm" name="customerForm"
        action="${pageContext.request.contextPath }/linkMan_findAll.action"
        method=post>

        <TABLE cellSpacing=0 cellPadding=0 width="98%" border=0>
            <TBODY>
                <TR>
                    <TD width=15><IMG src="${pageContext.request.contextPath
}/images/new_019.jpg"
                        border=0></TD>
                    <TD width="100%"
background="${pageContext.request.contextPath }/images/new_020.jpg"
                        height=20></TD>
                    <TD width=15><IMG src="${pageContext.request.contextPath
}/images/new_021.jpg"
                        border=0></TD>
                </TR>
            </TBODY>
        </TABLE>
        <TABLE cellSpacing=0 cellPadding=0 width="98%" border=0>
            <TBODY>
                <TR>
                    <TD width=15 background="${pageContext.request.contextPath
}/images/new_022.jpg"><IMG
                        src="${pageContext.request.contextPath
}/images/new_022.jpg" border=0></TD>
                    <TD vAlign=top width="100%" bgColor=#ffffff>
```

```
                        <TABLE cellSpacing=0 cellPadding=5 width="100%"
border=0>
                            <TR>
                                <TD class=manageHead>当前位置：联系人管理 &gt; 联系人
列表</TD>
                            </TR>
                            <TR>
                                <TD height=2></TD>
                            </TR>
                        </TABLE>
                        <TABLE borderColor=#cccccc cellSpacing=0 cellPadding=0
                            width="100%" align=center border=0>
                            <TBODY>
                                <TR>
                                    <TD height=25>
                                        <TABLE cellSpacing=0 cellPadding=2
border=0>
                                            <TBODY>
                                                <TR>
                                                    <TD>联系人名称：</TD>
                                                    <TD><INPUT class=textbox
id=sChannel2
                                                        style="WIDTH: 80px"
maxLength=50 name="lkmName"></TD>

                                                    <TD><INPUT class=button
id=sButton2 type=submit
                                                        value=" 筛选 "
name=sButton2></TD>
                                                </TR>
                                            </TBODY>
                                        </TABLE>
                                    </TD>
                                </TR>

                                <TR>
                                    <TD>
                                        <TABLE id=grid
                                            style="BORDER-TOP-WIDTH: 0px; FONT-
WEIGHT: normal; BORDER-LEFT-WIDTH: 0px; BORDER-LEFT-COLOR: #cccccc; BORDER-
BOTTOM-WIDTH: 0px; BORDER-BOTTOM-COLOR: #cccccc; WIDTH: 100%; BORDER-TOP-COLOR:
#cccccc; FONT-STYLE: normal; BACKGROUND-COLOR: #cccccc; BORDER-RIGHT-WIDTH: 0px;
TEXT-DECORATION: none; BORDER-RIGHT-COLOR: #cccccc"
                                            cellSpacing=1 cellPadding=2
rules=all border=0>
                                            <TBODY>
                                                <TR
                                                    style="FONT-WEIGHT: bold;
FONT-STYLE: normal; BACKGROUND-COLOR: #eeeeee; TEXT-DECORATION: none">
                                                    <TD>联系人名称</TD>
                                                    <TD>性别</TD>
                                                    <TD>办公电话</TD>
                                                    <TD>手机</TD>
                                                    <TD>邮箱</TD>
                                                    <TD>QQ</TD>
                                                    <TD>职位</TD>
                                                    <TD>所属客户</TD>
                                                    <TD>操作</TD>
```

```
                                                </TR>
                                                <s:iterator value="list">
                                                <TR
                                                    style="FONT-WEIGHT: normal;
FONT-STYLE: normal; BACKGROUND-COLOR: white; TEXT-DECORATION: none">
                                                    <TD><s:property
value="lkm_name"/></TD>
                                                    <TD><s:property
value="lkm_gendar"/></TD>
                                                    <TD><s:property
value="lkm_phone"/></TD>
                                                    <TD><s:property
value="lkm_mobile"/></TD>
                                                    <TD><s:property
value="lkm_email"/></TD>
                                                    <TD><s:property
value="lkm_qq"/></TD>
                                                    <TD><s:property
value="lkm_position"/></TD>
                                                    <TD><s:property
value="customer.cust_name"/></TD>

                                                    <TD>
                                                    <a
href="${pageContext.request.contextPath }/">修改</a>
                                                      
                                                    <a
href="${pageContext.request.contextPath }/">删除</a>
                                                    </TD>
                                                </TR>

                                                </s:iterator>

                                            </TBODY>
                                        </TABLE>
                                    </TD>
                                </TR>

                                <TR>
                                    <TD><SPAN id=pagelink>
                                            <DIV
                                                style="LINE-HEIGHT: 20px;
HEIGHT: 20px; TEXT-ALIGN: right">
                                                共[<B><s:property
value="totalCount"/></B>]条记录,[<B><s:property value="totalPage"/></B>]页
                                                ,每页显示

                                                <select name="pageSize"
onchange="to_page()">
                                                    <option value="3" <s:if
test="pageSize == 3">selected</s:if>>3</option>
                                                    <option value="5" <s:if
test="pageSize == 5">selected</s:if>>5</option>
                                                    <option value="10" <s:if
test="pageSize == 10">selected</s:if>>10</option>
                                                </select>
                                                条
                                                <s:if test="currPage != 1">
```

```
                                                    [<A
href="javascript:to_page(<s:property value="1" />)">首页</A>]
                                                    [<A
href="javascript:to_page(<s:property value="currPage-1" />)">前一页</A>]
                                                </s:if>  
                                                <B>

                                                    <s:iterator var="i"
begin="1" end="totalPage">
                                                        <s:if test="#i ==
currPage">
                                                            <s:property
value="#i"/>
                                                        </s:if>
                                                        <s:else>
                                                            <a
href="javascript:to_page(<s:property value="#i" />)"><s:property value="#i"/>
</a>
                                                        </s:else>
                                                    </s:iterator>

                                                </B>  
                                                <s:if test="currPage !=
totalPage">
                                                    [<A
href="javascript:to_page(<s:property value="currPage+1" />)">后一页</A>]
                                                    [<A
href="javascript:to_page(<s:property value="totalPage" />)">尾页</A>]
                                                </s:if>
                                                到
                                                <input type="text" size="3"
id="page" name="currPage" />
                                                页

                                                <input type="button" value="Go"
onclick="to_page()"/>
                                            </DIV>
                                        </SPAN></TD>
                                    </TR>
                                </TBODY>
                            </TABLE>
                        </TD>
                        <TD width=15 background="${pageContext.request.contextPath
}/images/new_023.jpg"><IMG
                            src="${pageContext.request.contextPath
}/images/new_023.jpg" border=0></TD>
                    </TR>
                </TBODY>
            </TABLE>
            <TABLE cellSpacing=0 cellPadding=0 width="98%" border=0>
                <TBODY>
                    <TR>
                        <TD width=15><IMG src="${pageContext.request.contextPath
}/images/new_024.jpg"
                            border=0></TD>
                        <TD align=middle width="100%"
                            background="${pageContext.request.contextPath
}/images/new_025.jpg" height=15></TD>
```

```
                    <TD width=15><IMG src="${pageContext.request.contextPath
}/images/new_026.jpg"
                        border=0></TD>
                </TR>
            </TBODY>
        </TABLE>
    </FORM>
</BODY>
</HTML>
```