Noka Laboratory

Noka of China Co., Ltd.

BS 研究中心

BS Research Center

Noka Pro 开发手册

Noka Pro Development Manual

文档类别	技术手册
版　　本	V3.4
密　　级	不涉密
起 草 人	谢方建
审核部门	
确认部门	

二〇一四年十一月

使用申明

本软件知识产权归诺亚(中国)开源科技所有，Noka Pro 为商业软件仅限于以学习或研究为目的使用，不能以任何形式使用或嵌入到其它商业软件内部使用。

本软件作为免费软件发布，不对其软件的使用承担任何责任，如有需要以商业软件发布或技术支持欢迎咨询具体发布事宜

Noka Laboratory
Rebin

网址：http://www.97521.com
电话：(028)85145708，85143878，85147698
电邮：xiefangjian@163.com

目录

网址：http://www.97521.com
电话：(028)85145708，85143878，85147698
电邮：xiefangjian@163.com

第1章 Noka Pro 部署

1.1 Noka Pro 3.X 新特性

Noka Pro 3.X是基于Noka Tag 6.0研发的企业信息系统快速开发平台，集成了系统管理基础功能（包括基础字典、树形字典、权限管理、个人信息）可以快速组建任何企业信息系统所需要的基础架构。

Noka Pro 3.X第三方支撑平台

名称	版本	备注
JDK	1.7.0(以上)	
Tomcat	5.5(以上)	需要支持jsp2.0标准
WebLogic	9.2(以上)	
resin	4.0(以上)	

Noka Pro 3.X支持的数据列表。

数据库	版本	备注
SQL Server	2000	
SQL Server	2005	
Oracle	9i(以上)	
MySQL	5.0(以上)	
PostgreSQL	7.0(以上)	
DB2	9.7(以上)	

Noka Pro 3.X 浏览器兼容列表

浏览器名称	版本	备注
Internet Explorer	6.0(以上)	
Mozilla Firefox	2.0(以上)	
Netscape Navigator		
Opera		
Apple Safari		
Google Chrome		

Noka Pro 3.X 技术架构清单

技术架构名称	版本	备注
Struts2	2.3.4.1	
hibernate	3.0	
proxool	0.9.0	
Apache fop	1.0	
Noka tag	6.0.9	

1.2 Noka Pro 配置

Noka Pro 3.X 的配置非常简单，具体步骤如下：

1、在db目录下选择你所熟悉的数据库所对应的目录，执行里面的创建表sql语句，然后在执行inisql目录下的系统初始化sql语句。如果没有你需要的sql脚本请用PowerDesigner打开pord目录下的文件生成你需要的数据库sql脚本。

2、用你所熟悉的开发工具新建一个Java Web项目，将src下的src和WebRoot拷贝到你所新建的工程里面覆盖相对应的目录。src为java源代码目录，WebRoot为jsp源代码目录，某些开发工具jsp源代码目录可能不是名为WebRoot的目录，需要自行更改。

3、修改src目录或是WebRoot\WEB-INF\classas目录下的proxool.xml文件，该文件为proxool数据库连接池的配置文件，Noka Pro将会使用该配置，有关proxool.xml的详细配置请参考proxool官方手册。

4、配置完成以后，就可以发布你的项目到相应的应用服务器上进行测试，系统初始登录用户名和密码都为admin。

5、在src目录下的system_config.xml文件里面，是noka pro的系统配置文件，内容看起来可能会是下面这样的：

```
<?xml version="1.0" encoding="UTF-8"?>
<xmlconst>
<option key="skin"><![CDATA[skins/orange.nks]]></option><!-- 项目皮肤  -->
```


网址：http://www.97521.com
电话：(028)85145708，85143878，85147698
电邮：xiefangjian@163.com

```
<!-- 数据列表每面显示记录总数  -->
<option key="dbtable-page-size"><![CDATA[10]]></option>
<option key="default-language"><![CDATA[zh-cn]]></option><!-- 默认语言 -->
<option key="path-language"><![CDATA[language]]></option><!--语文件路径 -->
<!--排除 Session 验证的路径,以逗号分隔 -->
<option key="no-sessionCheck">
<![CDATA[login.nk,index.jsp,codeimg_nsk.codeimg,noka.ico]]>
</option>
<!--需要 session 才能访问的共公路径 -->
<option
key="session-publick"><![CDATA[login.nk,main.nk,top.nk,nkwordDbdb.ndbgrid,nkwordDbd
bbn.ndbgrid,left.nk,middle.nk,right.nk,rightmenu.nk,bottom.nk,mypasswordupdate.nk,m
ypassword.nk,ntreeselect_*.tre]]></option>
<!--session 验证失败重定向路径 -->
<option key="no-sessionUrl"><![CDATA[${rooturl}/login.nk]]></option>
<!—sql 防注入验证失败重定向路径 -->
<option key="no-sqlUrl"><![CDATA[${rooturl}/login.nk]]></option>
<option key="cache-file"><![CDATA[true]]></option><!--缓存文件 -->
<!-- 数据过滤  key 指过滤 sql 的 key 一般可指表名,user-field 用户 id 字段,dept-field 用
户所在部门字段 lan 语言文件的 key select 对应于数据过滤设置界面上的 5 个选项，为 false
表示不显示该选项 -->
<dbtable-filters>
    <table key="NK_SYS_WORDBOOK" user-field="WUSER" dept-field="WCWORKDEPT"
lang="org.noka.sys.worktitle"></table>
    <table key="NK_SYS_USERINFO" user-field="USID" dept-field="USWORK"
lang="org.noka.sys.usertitle" select="true,true,true,false,true"></table>
</dbtable-filters>
<!— fop 打印接口配置，建议废弃,用 noka-tag 的自定义接口实现 -->
<prints>
    <print id="test" class="org.noka.printItem.TestPrint"></print>
</prints>
<!-- 浏览器文件缓存控制 D 表示天，h 表示小时,m 表示秒,s 表示秒-->
<cache-files>
<cache-file extension="*.js" expires="1S"></cache-file>
<cache-file extension="*.css" expires="30h"></cache-file>
<cache-file extension="*.jpg" expires="30m"></cache-file>
<cache-file extension="*.png" expires="30s"></cache-file>
<cache-file extension="*.gif" expires="30D"></cache-file>
</cache-files>
</xmlconst>
```


网址：http://www.97521.com
电话：(028)85145708，85143878，85147698
电邮：xiefangjian@163.com

第2章 Noka Pro 开发入门

2.1 NokaPro 目录规范说明

1、src（源码）目录约定规范说明

目录名称约定	作用
org.noka.action	Struts的Action类所在目录
org.noka.constvar	系统全局常量/变量类所在目录
org.noka.filter	Noka Pro系统拦截器所在目录
org.noka.function	系统级公共方法类所在目录
org.noka.item	数据库实体类及其实体类配置文件所在目录
org.noka.print	fop打印接口实现类所在目录
org.noka.printitem	fop打印自定义业务类及其打印模版所在目录
org.noka.pro	系统角色分配异步请求调用方法类所在目录
org.noka.socketserver	系统内置的socket服务实现类
org.noka.test	Noka-tag dbgrid自定义导出/打印接口实现类实例
org.noka.tlg	系统级自用标签库实现类所在目录

2、WEB（展现层）目录约定规范说明

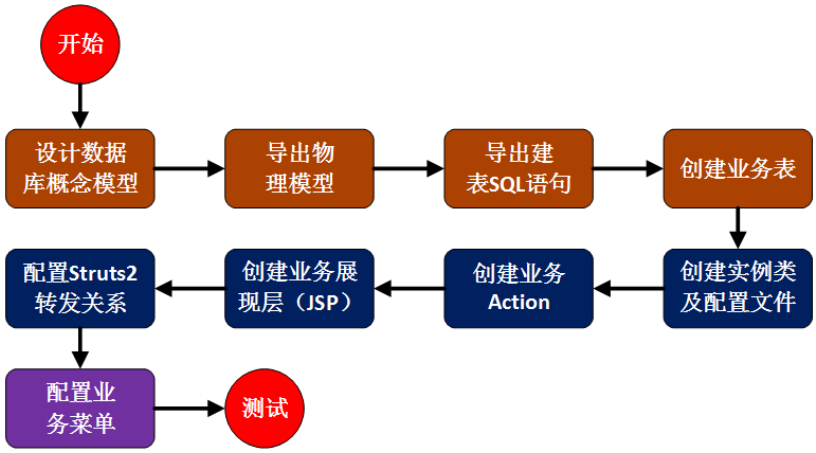
目录名称约定	作用
inc	系统公共引用JSP所在目录
language	系统语言文件所地目录
main	系统登录界面及主界面所在目录
peopselect	系统级公用用户选择界面所在目录
popedom	用户及角色权限管理界面所在目录
script	系统javascript脚本所在目录
skins	系统皮肤文件所在目录
treebook	树形字典管理界面所在目录
user	系统当前用户修改自己密码及其基础信息界面所在目录
wordbook	基础字典管理界面所在目录

2.2 典型数据库操作功能开发实例

数据库典型的开发实例包括对数据表的增、删、改、查四大功能，Noka Pro

是基于Struts2研发的基础二次开发平台，因此所有功能的研发应当遵循Struts2的各项编码规则，本平台采用两层架构以减少代码的复杂程度，并在一定程度提高系统响应性能。

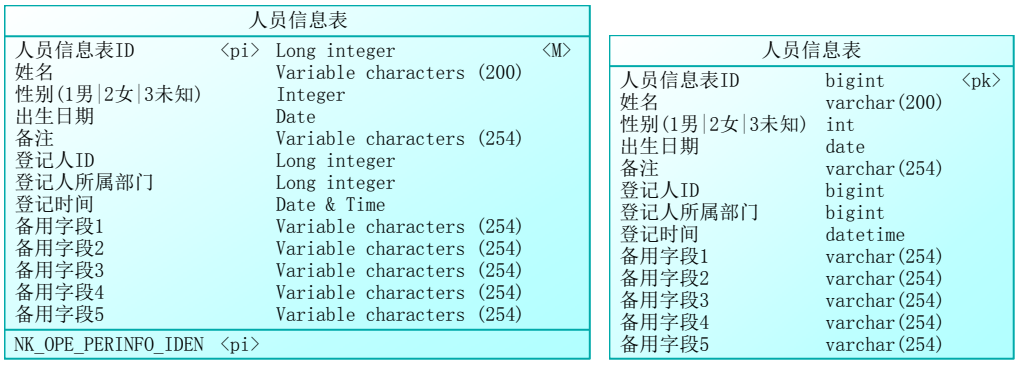
Noka Pro开发一般遵循以下开发步骤，其中部分步骤可以根据个人习惯前后调换，但在开发过程中必须遵循后面的编码规则，否则不确保整个系统的高性能及其稳定性。



（图2.2-01 一般开发流程图）

Step 1

根据流程图我们第一步选设计业务所需要的数据表并建好数据库表，在本实例中我们假定需要一个人员信息管理这样一个功能，既对人员信息进行增加、删除、修改、查询等功能，我们所涉及的人员信息表如下：



（图2.2-02 人员信息表概念模型及其物理模型）

网址：http://www.97521.com
电话：(028)85145708，85143878，85147698
电邮：xiefangjian@163.com

在设计数据库表时预留几个备用字段是一个比较好的习惯，便于业务扩展，另外在Noka Pro里数据表有三个字段在设计时建议作为必须字段，这三个字段为操作人ID，操作人所在部门ID，操作时间，上图所示的实例中对应的为登记人ID，登记人所属部门，登记时间，关于更多Noka Pro数据表设计规范详见本手册附件。

Step 2

建立数据表及其实体类和对应的配置文件，这里的我们以MySQL为例，对应的实体类及其hibernate配置文件建议用第三方小工具自动生成，本例的人员信息表实体类及其配置文件如下：

```
//1、实体类，注意：所有实体类都应该是可被序列化的
package org.noka.item;
import java.util.Date;
// data table name:nk_ope_perinfo
public class PerinfoItem implements  Serializable{
    private static final long serialVersionUID = 1879586942311284895L;
    private Long piid = null;// PIID BIGINT
    private String piname = null;// PINAME VARCHAR
    private Integer pisex = null;// PISEX INTEGER
    private Date pibirth = null;// PIBIRTH DATE
    private String pitext = null;// PITEXT VARCHAR
    private Long piwuid = null;// PIWUID BIGINT
    private Long piwudept = null;// PIWUDEPT BIGINT
    private Date piwdate = null;// PIWDATE DATETIME
    private String pialt1 = null;// PIALT1 VARCHAR
    private String pialt2 = null;// PIALT2 VARCHAR
    private String pialt3 = null;// PIALT3 VARCHAR
    private String pialt4 = null;// PIALT4 VARCHAR
    private String pialt5 = null;// PIALT5 VARCHAR
    public Long getPiid() {
        return piid;
    }
    public void setPiid(Long piid) {
        this.piid = piid;
    }
    public String getPiname() {
        return piname;
```

```
    }
    public void setPiname(String piname) {
        this.piname = piname;
    }
    public Integer getPisex() {
        return pisex;
    }
    public void setPisex(Integer pisex) {
        this.pisex = pisex;
    }
    public Date getPibirth() {
        return pibirth;
    }
    public void setPibirth(Date pibirth) {
        this.pibirth = pibirth;
    }
    public String getPitext() {
        return pitext;
    }
    public void setPitext(String pitext) {
        this.pitext = pitext;
    }
    public Long getPiwuid() {
        return piwuid;
    }
    public void setPiwuid(Long piwuid) {
        this.piwuid = piwuid;
    }
    public Long getPiwudept() {
        return piwudept;
    }
    public void setPiwudept(Long piwudept) {
        this.piwudept = piwudept;
    }
    public Date getPiwdate() {
        return piwdate;
    }
    public void setPiwdate(Date piwdate) {
        this.piwdate = piwdate;
    }
    public String getPialt1() {
```


网址：http://www.97521.com
电话：(028)85145708，85143878，85147698
电邮：xiefangjian@163.com

```
        return pialt1;
    }
    public void setPialt1(String pialt1) {
        this.pialt1 = pialt1;
    }
    public String getPialt2() {
        return pialt2;
    }
    public void setPialt2(String pialt2) {
        this.pialt2 = pialt2;
    }
    public String getPialt3() {
        return pialt3;
    }
    public void setPialt3(String pialt3) {
        this.pialt3 = pialt3;
    }
    public String getPialt4() {
        return pialt4;
    }
    public void setPialt4(String pialt4) {
        this.pialt4 = pialt4;
    }
    public String getPialt5() {
        return pialt5;
    }
    public void setPialt5(String pialt5) {
        this.pialt5 = pialt5;
    }
}
//2、实体类配置文件
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE hibernate-mapping PUBLIC
    "-//Hibernate/Hibernate Mapping DTD 3.0//EN"
    "http://hibernate.sourceforge.net/hibernate-mapping-3.0.dtd">
<hibernate-mapping package="org.noka.item">
<class name="PerinfoItem" table="NK_OPE_PERINFO">
<id name="piid">
    <generator class="increment"/>
</id>
    <property name="piname"/>
```


网址：http://www.97521.com
电话：(028)85145708，85143878，85147698
电邮：xiefangjian@163.com

```
    <property name="pisex"/>
    <property name="pibirth"/>
    <property name="pitext"/>
    <property name="piwuid"/>
    <property name="piwudept"/>
    <property name="piwdate"/>
    <property name="pialt1"/>
    <property name="pialt2"/>
    <property name="pialt3"/>
    <property name="pialt4"/>
    <property name="pialt5"/>
</class>
</hibernate-mapping>
//3、hibernate 注册实体类
<mapping resource="org/noka/item/PerinfoItem.hbm.xml"/>
```

Step 3

创建业务Action，我们的业务Action主要涉及数据展现、增加、删除、修改、查询五个功能，其中查询交由noka-tag标签自动完成，Action里面只需要实现前四个功能即可，下面来看我们的Action类实例：

```
package org.noka.action;
import java.io.PrintStream;
import java.util.ArrayList;
import java.util.Date;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import org.apache.struts2.ServletActionContext;
import org.noka.function.LanguageRead;
import org.noka.item.PerinfoItem;
import org.nokatag.system.DataUtil;

/**
 * 人员信管理实体类 所有 Action 类名必须以 Action 结尾，必须继承 BaseAction
 * @author rebin
 * @create 2014-11-20
 */
public class PerinfoAction extends BaseAction{
```


网址：http://www.97521.com
电话：(028)85145708，85143878，85147698
电邮：xiefangjian@163.com

```
    private static final long serialVersionUID = -1896304934938724952L;
    private PerinfoItem perinfo=null;//人员信息实体类
    private List<String> pids = null;//人员信息表记录 id
    //============================================================
    /**
     * 人员信显示界面
     * @return
     */
    public String perinfoForJsp(){
        HttpServletRequest request = ServletActionContext.getRequest();
        //如果需要多语言支持，侧通过向 LanguageRead.getLang()方法传入 Languag 文件中
的 key 返回对应语言的内容（自动根据浏览器语言返回）
        //String SQL="SELECT PIID,PINAME AS
"+LanguageRead.getLang("org.perinfo.jsp.name")+",PISEX AS
"+LanguageRead.getLang("org.perinfo.jsp.sex")+",PIBIRTH AS
"+LanguageRead.getLang("org.perinfo.jsp.birth")+",PITEXT AS
"+LanguageRead.getLang("org.perinfo.jsp.text")+",PISEX  FROM NK_OPE_PERINFO";
        String SQL="SELECT PIID,PINAME AS 姓名,PISEX AS 性别,PIBIRTH AS 出生日
期,PITEXT AS 备注,PISEX   FROM NK_OPE_PERINFO";//给前 noka dbgrid 的 SQL，不必担心安
全问题，所有 sql 均是加密的
        request.setAttribute("sql",  sqlFilter(SQL, "NK_OPE_PERINFO", request));//
标准写法，该方法在作数据过滤权限时起作用
        return "per_perinfo.jsp";//返回到前端的 jsp 页面
    }
    /**
     * 添加人员信息
     * @return
     */
    public String perinfoAdd(){
        HttpServletResponse response = ServletActionContext.getResponse();
        try{
            PrintStream out = new PrintStream(response.getOutputStream());//获取 out
输出对象
            if(perinfo!=null){
                perinfo.setPiwdate(new Date());//写入时间为当前系统时间
                DataUtil.insert(perinfo);
                 out.println("1");//添加成
            }else{
                out.println("2");//添加失败
            }
```


网址：http://www.97521.com
电话：(028)85145708，85143878，85147698
电邮：xiefangjian@163.com

```
        }catch(Exception se){}
        return null;
    }
    /**
     * 批量删除人员信息
     * @return
     */
    @SuppressWarnings({ "unchecked", "rawtypes" })
    public String perinfoDel(){
        HttpServletResponse response = ServletActionContext.getResponse();
        List perinfoList = new ArrayList();
        try{
            PrintStream out = new PrintStream(response.getOutputStream());//获取 out
输出对象
        if(pids!=null){
            for(int i=0;i<pids.size();i++){//前端传过来的为一组记录 ID,作批量删除
                PerinfoItem perinfoitem = new PerinfoItem();
                Long piid = Long.parseLong(pids.get(i));
                perinfoitem.setPiid(piid);
                perinfoList.add(perinfoitem);
            }
            if(!perinfoList.isEmpty()){
                DataUtil.deleteList(perinfoList);
                out.println("5");//删除成功
            }else{
                out.println("6");//删除失败
            }
        }
        }catch(Exception se){

        }
        return null;
    }
    /**
     * 修改人员信息
     * @return
     */
    public String perinfoUpdate(){
        HttpServletResponse response = ServletActionContext.getResponse();
        try{
        PrintStream out = new PrintStream(response.getOutputStream());//获取 out 输
```


网址：http://www.97521.com
电话：(028)85145708，85143878，85147698
电邮：xiefangjian@163.com

```
出对象
        if(perinfo!=null && perinfo.getPiid()!=null){
            DataUtil.update(perinfo);
            out.println("3");//修改成功
        }else{
            out.println("4");//修改失败
        }
        }catch(Exception se){}
        return null;
    }
    //==============set and get var==============================
    public PerinfoItem getPerinfo() {
        return perinfo;
    }
    public void setPerinfo(PerinfoItem perinfo) {
        this.perinfo = perinfo;
    }
    public List<String> getPids() {
        return pids;
    }
    public void setPids(List<String> pids) {
        this.pids = pids;
    }
    //=========================================================
}
```

Step 4

Action完成以后，我们需要编写展现层的jsp页面，在nokapro里jsp页面控件由noka-tag代替struts的标签，每个jsp页面必须加载

<%@ include file="/inc/inc.jsp"%>（需要登录验证的）

或者是

<%@ include file="/inc/nosession_inc.jsp"%>（不需要登录验证的）

本实例的jsp页面如下所示：

```
<%@ page language="java"  pageEncoding="UTF-8"%>
<%@ include file="/inc/inc.jsp"%>
```


网址：http://www.97521.com
电话：(028)85145708，85143878，85147698
电邮：xiefangjian@163.com

```
<html>
  <head>
    <title>人员信息管理</title>
    <meta http-equiv="pragma" content="no-cache">
    <meta http-equiv="cache-control" content="no-cache">
    <meta http-equiv="expires" content="0">
    <meta http-equiv="keywords" content="keyword1,keyword2,keyword3">
    <meta http-equiv="description" content="This is my page">
<!—加载本页的 javascript 文件，建议所有 javascript 脚本采用外部加载的方式引入 -->
    <script type='text/javascript' src='${rooturl}/script/per/perinfo.js'></script>
  </head>
  <body>
 <!-- 表单 start-->
<n:form action="" method="post" id="perinfoform" name="perinfoform"
onsuccess="onsuccess" onfailure="onfailure" subfunction="nk_submitform">
<!—
 表单字段的名称前缀要与 action 里面的变量名称相对应，如该例的字段名称前缀为 perinfo
-->
<!— 隐藏字段 人员信息表 id，修改时使用—>
<input  type="hidden" name="perinfo.piid" id="pi_id" value="" />
<!— 隐藏字段 人员信息表登记人所属部门 id,修改/添加时使用,user 为全局对象直接使用-->
<input  type="hidden" name="perinfo.piwudept" value="${user.uswork}" />
<!— 隐藏字段 人员信息表登记人 id，修改/添加时使用,user 为全局对象直接使用-->
<input  type="hidden" name="perinfo.piwuid" value="${user.usid}" />
<table>
<tr><td>姓名：</td><td><n:InputText name="perinfo.piname" id="piname_id"
allownull="false" ondblclick="DoubleClickClear(this)"></n:InputText></td></tr>
<tr><td>性别：</td><td><n:Radios id="pisex_id" name="perinfo.pisex"
json="[{value:'1',label:'男'},{value:'2',label:'妇'},{value:'2',label:'未知'}]"
value="1"></n:Radios></td></tr>
<tr><td>出生日期：</td><td><n:date name="perinfo.pibirth" id="pibirth_id"
prave="dateFmt:'yyyy-MM-dd'"></n:date></td></tr>
<tr><td>备注：</td><td><n:Textarea name="perinfo.pitext"
id="pitext_id"></n:Textarea></td></tr>
<tr><td>
<!—
 系统按钮标签，id 为数据库菜单表中配置的按钮 ID，jsp 为本页的 jsp 名称，name 为按钮的名
字（等同于普通按钮的 name 属性）,menuid 本页面所在的菜单的 ID，muid 为系统变量直接使用
Classtyle 等同于普通按钮的 class 属性
Value 等同于普通按钮的 value 属性
Onclick 等同于普通按钮的 onclick 属性
```


网址：http://www.97521.com
电话：(028)85145708，85143878，85147698
电邮：xiefangjian@163.com

```
-->
  <b:button id="add_id" name="add" classstyle="but" onclick="addperinfo()"
value="${org.noka.sys.add}"/>  
  <b:button id="update_id" name="update" style="display: none;" classstyle="but"
onclick="updates()" value="${org.noka.sys.update}"/>
  </td></tr>
  <tr><td>
  <font style="color: red;" id="msg_util">${msg}</font>
  </td></tr>
  </table>
</n:form>
<n:DBGrid
    width="700"
    height="300"
    id="perinfoDB"
    onfailure="onfailure"
    onsuccess="onsuccess"
    sql="${sql}"
    selectfiled="姓名  like '%$[wdname,姓名]%'"
    selectinput="true"
    title="人员信息表${b:ButIf( 'add||update',org.noka.sys.dbcortitle)}"
    tableformname="dataform"
    checkname="pids"
    cells="width:100,show:4|width:200,show:1|width:200,show:1|width:60,show:1"
    checkd="checkbox"
    onrowdblclick="detail"
    pagesize="200"
    tableformid="id_dataform"
    tableformAction="perinfoDel.nk"
    compositor="PIWDATE"
    descorasc="desc"
    custombutton="${b:but('delete',org.noka.sys.dell,'dels()')}"
/>
  </body>
</html>
```

Step 5

编写完Action与jsp以需要进行struts配置，其中的增加、删除、修改是采用

Ajax异步提交的数据，因此在配置stuts时不需要返回参数，配置如下：

```
<!-- ================人员信管理======================================== -->
    <package name="perinfoadmin" extends="struts-default">
       <!-- 人员信息显示界面 -->
       <action name="perinfoForJsp" class="org.noka.action.PerinfoAction"
method="perinfoForJsp">
            <result name="per_perinfo.jsp">/per/perinfo.jsp</result>
        </action>
         <!—添加人员信息 -->
         <action name="perinfoAdd" class="org.noka.action.PerinfoAction"
method="perinfoAdd"></action>
         <!—删除人员信息 -->
         <action name="perinfoDel" class="org.noka.action.PerinfoAction"
method="perinfoDel"></action>
         <!—修改人员信息 -->
         <action name="perinfoUpdate" class="org.noka.action.PerinfoAction"
method="perinfoUpdate"></action>
    </package>
```

Step 6

为业务配置菜单，该步骤是将以上开发的功能配置到系统的菜单管理里面，在数据表NK_SYS_MENU中进行配置，该表是一个树形的表，系统只支持三级菜单，最后一级没有菜单图标，具体说明可参照NokaPro数据表设计文件。

配置完成菜单以后即可登录系统，先要权限管理里面将新配置的菜单赋予当前用户，刷新界面以后就可以看到新增加的功能。

第3章 Noka Pro 开发说明

3.1 Noka Pro 菜单配置说明

Noka Pro菜单配置以系统菜单表为主要配置对象，菜单在系统中分共分为三级，其中一、二级显示在左边以折叠式菜单展现出来，第三级菜单在右边以选项卡的方式展现出来。菜单配置表在NK_SYS_MENU表中，具体配置如下：

字段名称	说明
MENUID	菜单表ID，全表唯一，不可重复
MENUPID	菜单表的父ID，它与MENUID一起形成组成树状结构
MENUNAME	菜单的名称，这里写入的是lanuage中的key
MENUURL	菜单的URL地址，${rooturl}代表URL全路径
MENUOPERATE	该菜单下的按钮组
MENUIMAGE	菜单图标，只有二级菜单有用，需要用${skin}指代皮肤文件
MENUTAXIS	菜单在同级菜单中的显示先后顺序
MENUTARGET	菜单的连接目标，只有二级菜单有用，固定值mainFrame
MENUNBURL	该菜单下需要访问的公共路径，表示拥有该菜单权限即可访问
MENULEVE	该菜单在树形菜单中的等级关系

（1）MENUURL一般需建议写全路径，

三级菜单写法：如一个菜单的访问地址为abc.nk（注意Noka Pro的action后缀为nk），菜单ID为12，则该菜单的URL可写为${rooturl}/abc.nk。

二级菜单写法：二级菜单的URL固定为${rooturl}/rightmenu.nk。

一级菜单写法：一级菜单的URL可为空，只需要ID和名字即可。

（2）MENULEVE由菜单的ID和父ID组成树形菜单的等级关系，其值格式为：

父ID的MENULENE_本级ID，如果是一级菜单为固定值-1_0_本级ID。

3.2 Noka Pro 页面工具说明

Noka Pro页面工具主要为b标签系列工具，其它工具如struts标签、noka-tag标签、c标签详见相对应的官方文档。

（1）B标签，该标签为按钮标签，自身具有权限控制功能，除以下几个特殊属性以外，其它属性同普通按钮标签属性等同。

属性	是否必须	说明
classstyle	否	等同于普通按钮的class属性
disabled	否	等同于普通按钮的disabled属性，其值为disabled时生效

（2）lan标签，语言文件读取标签，根据浏览器语言自动选择，传入值为语言文件的KEY。

属性	是否必须	说明
key	是	语言文件中所配置的key
value	否	该属性为变量名称，如设置该值，lan标签将根据key所取得的内容以value所示的变量名称设置在request中，通过${变量名}可取得

（3）but方法，该方法为JSTL方法，写法如下：

```
${b:but('delete',org.noka.sys.dell,'dels()')}
```

第一个参数为按钮的自身的ID。

第二个参数为按钮的显示名称，这里传入的是语言文件的KEY。

第三个参数是按钮点击时调用的javascript方法。

（4）ButIf方法，该方法为JSTL方法，写法如下：

```
${b:ButIf( 'delete','false:true')}
```

第一个参数为按钮所在jsp页面的文件名称加检测值，其中检测值是指按钮在数据库里配置的标识，根据按钮标识检测是否有该按钮权限，支持多个条件，条件判断可以用&&表示并且，用||表示或者。如delte||add表示如果有删除或是添加

按返回对应的值

第二个参数为返回值，当第二个参数验证成功以后返回:号前面的，失败返回:后面的。

（5）在任何JSP页面上均可以通过${user}获取当前登录用户的详细信息。

3.3 Noka Pro 多语言配置说明

默认状态下，语言文件存放在language目录下，语言文件命名规则遵循struts国际化支持的命名规则，其名称为language_具体语言，如简体中文则为language_zh-CN.xml。

```
<page id="org.noka.sys" description="系统全局信息">
<option key="dbcortitle" description="双击修改"><![CDATA[[双击修改]]]></option>
</page>
```

Page节点是对language的分组，一般来说不同的分级具有不同的功能，通常建议一个功能分一个组。

Id是指该组的ID，全文件唯一，不可重复。

Description仅作说明使用无实际意义，可选。

Option为language的具体配置项。

Key是指待该分组下的语言配置key，包内唯一。

![CDATA[[实际的内容]]]

在任何地方获取多语言信息时，应当是包的id加上包内的key共同组成获取语言内容的key，如上例所示，应为org.noka.sys.dbcortitle。

第4章 附件

4.1 Noka Pro 编码规范

（2）数据库操作规范

A、在 Action 内部凡使用 hibernate 操作数据库的必须使用 noka-tag 的辅助类进行操作。

B、在 Action 内部或是其它通过 http 调用的 java 方法内部，直接操作数据库的，其数据库连接对象必须使用 noka-tag 的 ServletNokaContext 对象获取，使用完以后，不能关闭。

C、切记不能在嵌套、递归、循环内操作数据库。

D、格式化字段、数据码表查找等尽量根据业务需经，通过创建 SQL 方法完成。

4.2 Noka Pro 数据库设计规范

（1）数据库、数据表、字段命名一率采用全大写命名。

（2）数据库、数据表名称必须带 NK_前缀。

（3）数据表如果是业务表必须使用 NK_OPE_前缀，系统表必须使用 NK_SYS_前缀。

（4）数据表字段必须带表名前缀，前缀取表名前两个字母。

（5）所有数据表必须预留 5 个以上的备用字段，字段类型为 VARCHAR(254)。

（6）所有数据表必须设置三个系统字段，分别为数据录入人的 ID，数据录入人所在部门 ID，录入时间。

（7）所有表必须有一个以上的主键。

（8）数据库涉及码表的，一率使用数据字典或树形字典进行配置，不得在新建码表。