## 一、 项目本地导入

MyEclipse 新建 web project ， 覆盖对应 src 和 WebRoot

Eclipse 新建 Dynamic web project ， 覆盖src ， 将WebRoot 中内容复制 WebContent 目录

JavaEE 企业级应用软件，布局经常采用 Frameset ， 左侧菜单树使用Dztree js组件制作的

目标功能 ：

1、 登陆

2、 添加用户 （简历上传）

3、 组合条件 员工信息列表查询

4、 员工信息详情查看（简历下载）

5、 员工信息删除

6、 员工信息编辑

7． 异常处理

8. 登录校验

## 二、 数据库设计

#新建数据库

create database struts2exec;

#创建用户

create user struts2@localhost identified by 'struts2';

#授权

grant all on struts2exec.\* to struts2@localhost;

\*\*\*\*\* Oracle和MySQL 作为应用数据库区别

mysql存在数据库概念，在企业开发中，针对一个项目创建一个单独数据库，创建单独用户， 为用户授予数据库权限 ，

oracle 一个数据库就是一个服务，在这个库中可以存在很多用户，每个用户有单独表空间 ，针对一个项目，只需要创建一个用户

#用户表

CREATE TABLE S\_User(

userID INT NOT NULL AUTO\_INCREMENT, #主键ID

userName VARCHAR(50) NULL, #用户姓名

logonName VARCHAR(50) NULL, #登录名

logonPwd VARCHAR(50) NULL, #密码#

sex VARCHAR(10) NULL, #性别（例如：男，女）

birthday VARCHAR(50) NULL, #出生日期

education VARCHAR(20) NULL, #学历（例如：研究生、本科、专科、高中）

telephone VARCHAR(50) NULL, #电话

interest VARCHAR(20) NULL, #兴趣爱好（例如：体育、旅游、逛街）

path VARCHAR(500) NULL, #上传路径（path路径）

filename VARCHAR(100) NULL, #上传文件名称（文件名）

remark VARCHAR(500) NULL, #备注

PRIMARY KEY (userID)

);

#初始化数据：默认用户名和密码是admin

INSERT INTO s\_user (userID,userName,logonName,logonPwd) VALUES (1,'超级管理员','admin','admin');

## 三、 搭建开发环境

struts2 + javabean + DAO + C3P0 + DBUtils + MySQL

导入jar包 和 配置文件

创建包结构

cn.itcast.user.domain

cn.itcast.user.dao

cn.itcast.user.service

cn.itcast.user.web.action

cn.itcast.user.utils

## 四、 功能开发 ---- 登陆功能

内网应用系统通常不需要验证码技术

1、登陆页面 WebRoot/login/login.jsp

form元素 改造 <s:form>

\* form的action 提交路径 怎么写 ？

企业中为了Action 代码可维护性，通常一个业务模块（Model）使用 一个Action

将登陆、 员工增删改查 对应业务方法 写入到 同一个Action

2、登陆表单数据校验

用户名 非空，3-12位

密码 非空

3、完成登陆逻辑，session保存用户， 跳转主页

top.jsp ${user.userName } 显示当前登陆用户

## 五、 功能开发 ---- 员工添加

1、 员工添加页面 WebRoot/user/add.jsp

form元素 改造 <s:form>

日期输入，使用 jquery datapicker控件

\* $('#birthday').datepick({dateFormat: 'yy-mm-dd'});

<s:form action="user\_add"> 访问 UserAction 中 add方法

2、 数据校验

校验出现input跳转问题，登陆页面校验失败跳转 login.jsp , 员工添加校验失败跳转 add.jsp ---- 只能配置一个input

\* <interceptor name="workflow" class="com.opensymphony.xwork2.interceptor.DefaultWorkflowInterceptor"/> 完成跳转

可以 通过 @InputConfig注解，改为校验失败后 跳转视图

3、 上传简历

如果真实文件名保存，出现覆盖问题 ------------ 唯一文件名 UUID

\* 在DB保存 uuid文件名路径 和 真实文件名

## 六、 组合条件查询 -- 查询员工列表

1、 查询页面 WebRoot/user/list.jsp

form 改造 <s:form>

<s:form action="user\_list" > 提交请求 UserAction 的 list方法

2、 条件组合查询

SQL语句 动态拼接

String sql = "select \* from s\_user where 1=1 ";

List<String> argList = new ArrayList<String>(); // 参数列表

if (user.getUserName() != null

&& user.getUserName().trim().length() > 0) {

sql += "and userName like ? ";

argList.add("%" + user.getUserName() + "%");

}

if (user.getSex() != null && user.getSex().trim().length() > 0) {

sql += "and sex = ? ";

argList.add(user.getSex());

}

if (user.getEducation() != null

&& user.getEducation().trim().length() > 0) {

sql += "and education = ? ";

argList.add(user.getEducation());

}

if (user.getIsUpload() != null

&& user.getIsUpload().trim().length() > 0) {

if (user.getIsUpload().equals("1")) {

// 上传简历

sql += "and filename is not null";

} else if (user.getIsUpload().equals("2")) {

// 没有上传简历

sql += "and filename is null";

}

}

3、 Action将查询结果 定义成员变量，提供getter方法 （值栈）

4、 JSP 通过struts2 标签 显示数据

\* 修改指向条件查询链接

修改 left.jsp

d.add(3,2,'用户管理','${pageContext.request.contextPath}/user\_list.action','','mainFrame');

修改 struts.xml

<result name="addSUCCESS" type="redirectAction">user\_list</result>

<result name="listSUCCESS">/user/list.jsp</result>

## 七、 功能实现 --- 员工删除功能

list.jsp 点击每行数据后 删除图标，删除该行数据

删除原理： 根据id 删除

<s:a action="user\_delete" namespace="/">

<s:param name="userID" value="#user.userID"></s:param>

<img src="${pageContext.request.contextPath}/images/i\_del.gif" width="16" height="16" border="0" style="CURSOR: hand">

</s:a>

服务器根据 id 删除数据表记录

\*\* 同时删除员工的简历

JavaScript 确认删除效果

$(".delLink").click(function(event){

var isConfirm = window.confirm("想好了吗？");

if(!isConfirm){

// 阻止提交

event.preventDefault();

}

});

## 八、 员工信息详情查看

原理 ： 点击按钮，传递员工id，在服务器根据id查询， 将查询结果保存值栈，页面回显

1、查询按钮的改写

<s:a action="user\_view" namespace="/">

<s:param name="userID" value="#user.userID"></s:param>

<img src="${pageContext.request.contextPath}/images/button\_view.gif" border="0" style="CURSOR: hand">

</s:a>

2、 Action查询数据结果，将结果被 getModel 引用

\* 通过getModel将查询结果 传递jsp

3、 使用model传递结果数据给jsp ，并不是栈顶model对象

## 九、 员工简历下载

1、 改造页面链接

<s:a action="user\_download" namespace="/" cssClass="cl\_01">

<s:param name="userID" value="model.userID"></s:param>

<s:property value="model.filename"/>

</s:a>

传递下载简历员工id 给服务器

2、 服务器接收id，查询员工简历文件，提供下载

\* 一个流 ，两个头信息

bug ： 查看无简历 员工信息时，出现异常

org.apache.jasper.JasperException: Caught an exception while getting the property values of cn.itcast.user.web.action.UserAction@1264e3b - Class: ognl.OgnlRuntime File: OgnlRuntime.java Method: getMethodValue Line: 1440 - ognl/OgnlRuntime.java:1440:-1

原因 ：<s:debug/> ----- 调用值栈中所有 getXXX 方法

\*\*\*\*\*\*\*\*\*\* 解决：1） 不用 <s:debug/> 2）处理所有getXXX方法可能出错的情况

## 十、 员工修改功能

原理 ：

流程一 根据id查询员工信息，通过值栈将员工信息 显示form表单中

1） 修改编辑按钮

<s:a action="user\_editview" namespace="/">

<s:param name="userID" value="#user.userID"></s:param>

<img src="${pageContext.request.contextPath}/images/i\_edit.gif" border="0" style="CURSOR: hand">

</s:a>

2） 通过model引用，将结果数据传递jsp

\*\*\*\*\* 编辑form 和 添加 form 基本一样， 区别编辑form 含有隐藏域，保存编辑员工id

流程二 提交修改form表单，完成员工信息修改

1） 修改员工信息，form表单 是 上传表单， 用户可以选择简历

如果用户选择了简历，在服务器端替换原有简历

如果用户没选择简历，忽略对简历的修改

2） 上传新简历，替换原有简历

存在简历无法删除问题 （因为有时简历被占用的 ）

解决 ： 如果不能及时删除简历， 可以在文件夹简历 简历文件删除标志文件

3） 校验

@InputConfig 定义返回 不同视图

## 十一、登陆校验拦截器

1、 自定义拦截器

获取Action对象添加错误信息

ActionSupport action = (ActionSupport) actionInvocation.getAction();

action.addActionError(action.getText("nologin"));

2、 配置拦截器

<interceptors>

<!-- 注册 -->

<interceptor name="login" class="cn.itcast.user.web.interceptor.LoginInterceptor"></interceptor>

<interceptor-stack name="myStack">

<interceptor-ref name="defaultStack"></interceptor-ref>

<interceptor-ref name="login"></interceptor-ref>

</interceptor-stack>

</interceptors>

<!-- 设置默认拦截器栈 -->

<default-interceptor-ref name="myStack"></default-interceptor-ref>

<!-- 如果未登陆 定义全局结果集 -->

<global-results>

<result name="login">/login/login.jsp</result>

</global-results>

\*\*\*\*\* 元素有顺序的！！！！！！！！

问题：永远无法登陆！！！！！

\*\*\*\* 通过拦截器参数设置，控制哪些方法 不被拦截 ---- 对user\_login 不进行拦截

## 十二、 异常处理

如果发生错误，企业中 会第一时间 将错误转换为 自定义异常 错误 ---- 抛出

1、 自定义异常

将出现通用异常，转换为 自定义异常

try {

...

} catch(){

throw 自定义异常;

}

2、 通常在拦截器 抓取自定义异常，记录日志， 提供友好页面

3、 自定义异常拦截，通常不拦截所有异常 ---- 通用异常页面

<global-exception-mappings>

<!-- 全局错误页面 -->

<exception-mapping result="error" exception="java.lang.Exception"></exception-mapping>

</global-exception-mappings>

\*\*\* 通用错误页面 可以 与 自定义异常错误页面 不同