###### 模块

权限模块

###### 功能

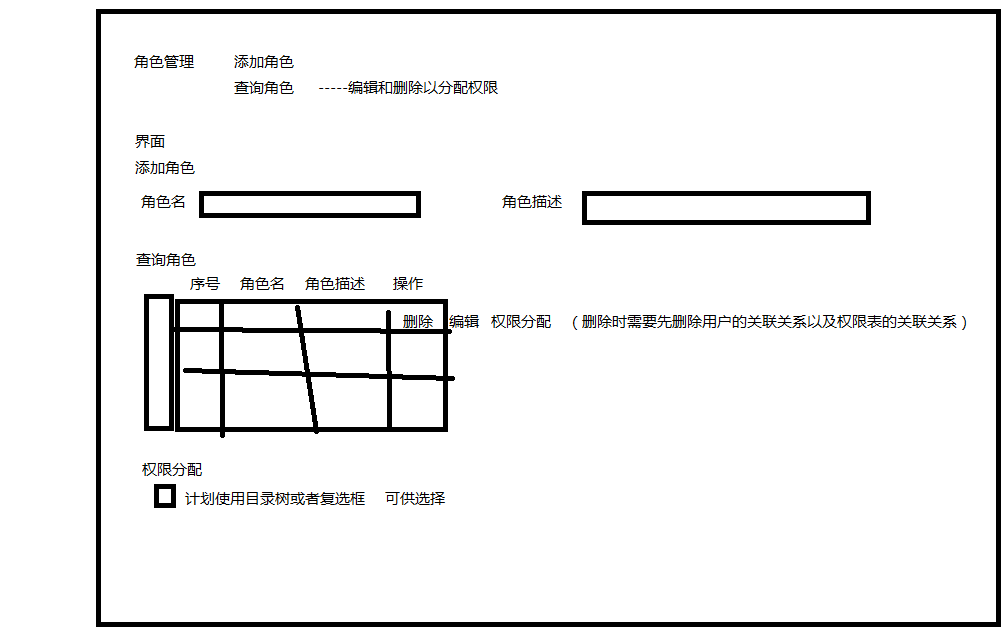
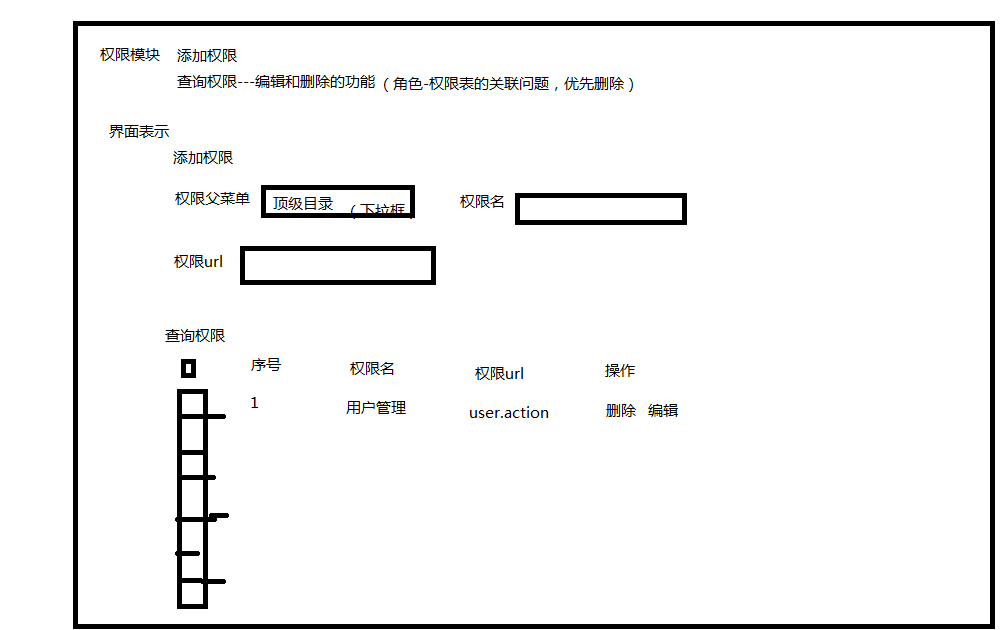
权限模块

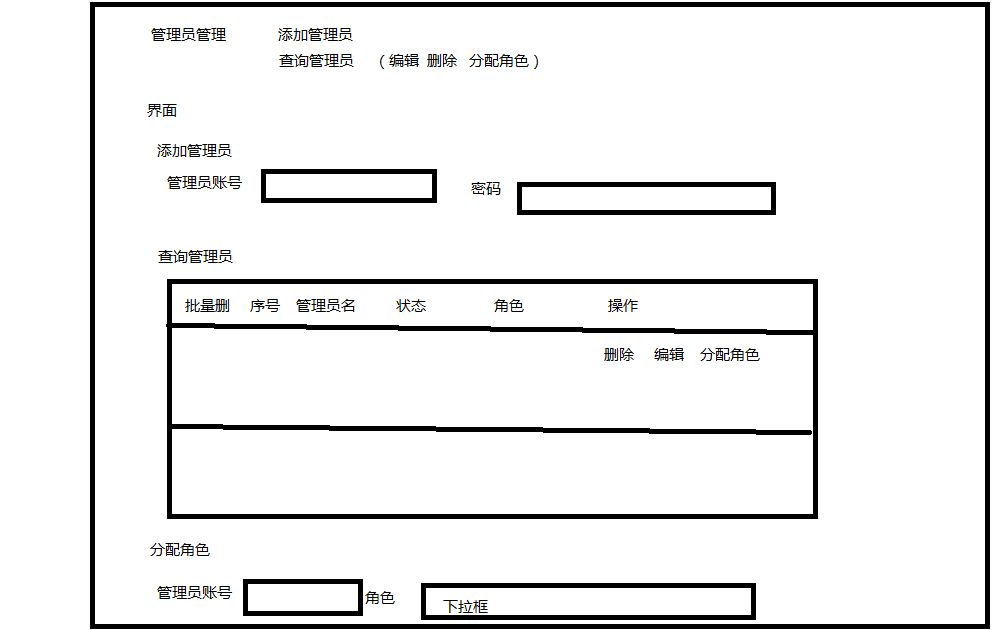


大概要实现的模型：

能画图就别说话

灵魂画师 素描10级证书





所以权限模块包含 3个子模块

管理员管理

角色管理

权限管理

预定义目录树为 3级菜单

**系统管理**

**管理员管理**

**添加管理员**

**查询管理员**

**角色管理**

**添加角色**

**查询角色**

**权限管理**

**添加权限**

**查询权限**

**平台管理**

**新闻管理**

**新闻栏目管理**

**新闻内容管理**

##### 开发过程中的问题

这部分，留下的每一句话，每一个问题都是关爱。

本以为j-ui是傻瓜式开发，做到的时候才发现，真的是傻瓜式开发，但是需要一定程度的阅读API才能做傻瓜。

拨出时间，阅读API。

###### 问题1

Spring mvc 拦截请求问题

遇到了这种问题。 关键是找不到北

Web.xml 里面配置

<servlet-mapping>

<servlet-name>dispatcher</servlet-name>

<!-- 拦截所有/rest/\* 的请求,交给DispatcherServlet处理,性能最好 -->

<url-pattern>/sys/\*</url-pattern>

</servlet-mapping>

刚开始

<url-pattern>/sys/\*</url-pattern>配置的是/ 就是说所有的请求都会被拦截，弄的所有的样式文件全部丢失

@RequestMapping(value = "/",method = RequestMethod.GET)

只有这里包含了 请求url，

<mvc:interceptors>

<mvc:interceptor>

<!--

/\*\*的意思是所有文件夹及里面的子文件夹

/\*是所有文件夹，不含子文件夹

/是web项目的根目录

-->

<mvc:mapping path=*"/"* />

<mvc:mapping path=*"/sys/\*\*"* />

<bean class=*"com.fr.here.interceptor.CommonInterceptor"*></bean>

</mvc:interceptor>

<!-- 当设置多个拦截器时，先按顺序调用preHandle方法，然后逆序调用每个拦截器的postHandle和afterCompletion方法 -->

</mvc:interceptors>

中配置的拦截才会生效

**注意三个地方配置的真正意思，这不是坑，这特么是井**

1. Controller里面

@RequestMapping(value = "/sys",method = RequestMethod.GET)

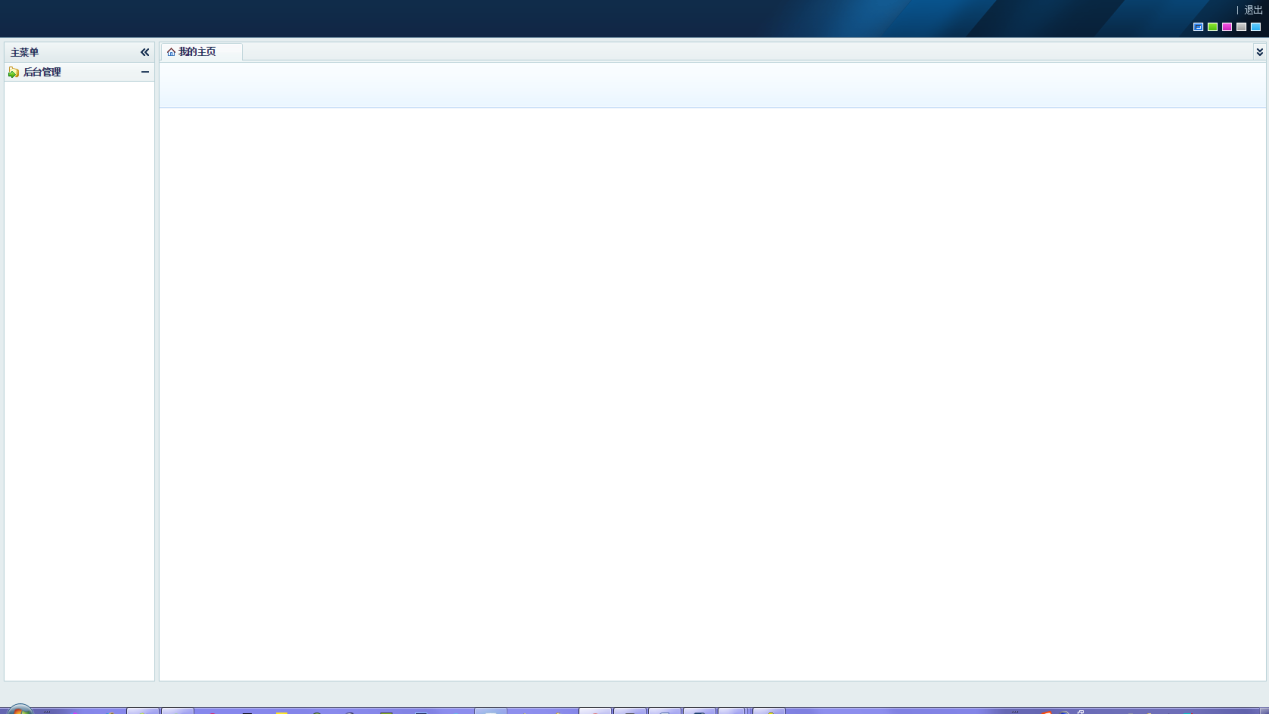
2. web.xml里面

<url-pattern>/rest/\*</url-pattern>

3.spring-mvc.xml里面

<mvc:interceptor>

<mvc:mapping path="/sys/\*\*" />



###### 问题2

同时多个项目使用Log4j打印日志，出现webAppRootKey异常

<!-- 配置log4j配置文件路径 -->

<context-param>

<param-name>log4jConfigLocation</param-name>

<param-value>classpath:log4j.properties</param-value>

</context-param>

<!-- 60s 检测日志配置 文件变化 -->

<context-param>

<param-name>log4jRefreshInterval</param-name>

<param-value>60000</param-value>

</context-param>

<context-param>

<param-name>webAppRootKey</param-name>

<param-value>fr-here.root</param-value>

指定唯一的webAppRootKey即可

###### 问题3

sprinMVC 怎么传参数，前台的参数又是怎么传递进来的

**Spring接收请求参数:**

**1,使用HttpServletRequest获取**

**Java代码  收藏代码**

1. @RequestMapping("/login.do")
2. **public** String login(HttpServletRequest request){
3. String name = request.getParameter("name")
4. String pass = request.getParameter("pass")
5. }

**2,Spring会自动将表单参数注入到方法参数，和表单的name属性保持一致。和Struts2一样**

**Java代码  收藏代码**

1. @RequestMapping("/login.do")
2. **public** String login(HttpServletRequest request,
3. String name,
4. @RequestParam("pass")String password) // 表单属性是pass,用变量password接收
5. {
6. syso(name);
7. syso(password)
8. }

**3,自动注入Bean属性**

**Java代码  收藏代码**

1. <form action="login.do">
2. 用户名：<input name="name"/>
3. 密码：<input name="pass"/>
4. <input type="submit" value="登陆">
5. </form>
7. //封装的User类
8. **public** **class** User{
9. **private** String name;
10. **private** String pass;
11. }

**Java代码  收藏代码**

1. @RequestMapping("/login.do")
2. **public** String login(User user)
3. {
4. syso(user.getName());
5. syso(user.getPass());
6. }

**向页面传值：**

**当Controller组件处理后，向jsp页面传值，**

**1，使用HttpServletRequest 和 Session  然后setAttribute()，就和Servlet中一样**

**2，使用ModelAndView对象**

**3，使用ModelMap对象**

**4，使用@ModelAttribute注解**

**Model数据会利用HttpServletRequest的Attribute传值到success.jsp中**

**Java代码  收藏代码**

1. @RequestMapping("/login.do")
2. **public** ModelAndView  login(String name,String pass){
3. User user = userService.login(name,pwd);
4. Map<String,Object> data = **new** HashMap<String,Object>();
5. data.put("user",user);
6. **return** **new** ModelAndView("success",data);
7. }

**使用ModelMap参数对象示例:**

**ModelMap数据会利用HttpServletRequest的Attribute传值到success.jsp中**

**Java代码  收藏代码**

1. @RequestMapping("/login.do")
2. **public**　String login(String name,String pass ,ModelMap model){
3. User user  = userService.login(name,pwd);
4. model.addAttribute("user",user);
5. model.put("name",name);
6. **return** "success";
7. }

**使用@ModelAttribute示例**

**在Controller方法的参数部分或Bean属性方法上使用**

**@ModelAttribute数据会利用HttpServletRequest的Attribute传值到success.jsp中**

**Java代码  收藏代码**

1. @RequestMapping("/login.do")
2. **public** String login(@ModelAttribute("user") User user){
3. //TODO
4. **return** "success";
5. }
7. @ModelAttribute("name")
8. **public** String getName(){
9. **return** name;
10. }

**Session存储：**

**可以利用HttpServletReequest的getSession()方法**

**Java代码  收藏代码**

1. @RequestMapping("/login.do")
2. **public** String login(String name,String pwd
3. ModelMap model,HttpServletRequest request){
4. User user = serService.login(name,pwd);
5. HttpSession session = request.getSession();
6. session.setAttribute("user",user);
7. model.addAttribute("user",user);
8. **return** "success";
9. }

**Spring MVC 默认采用的是转发来定位视图，如果要使用重定向，可以如下操作**

**1，使用RedirectView**

**2，使用redirect:前缀**

**Java代码  收藏代码**

1. **public** ModelAndView login(){
2. RedirectView view = **new** RedirectView("regirst.do");
3. **return** **new** ModelAndView(view);
4. }

**或者用如下方法，工作中常用的方法：**

**Java代码  收藏代码**

1. **public** String login(){
2. //TODO
3. **return** "redirect:regirst.do";
4. }

其中**使用@ModelAttribute这个方法获取参数的方法是个坑，应避免。**

###### 问题4

jstl标签使用

<c:forEach items="${privileges}" var=*"item"*>

<option value=*"*${item.id}*"*<c:if test="${model.pid == item.id}">selected='selected'</c:if>>${item.name}</option>

</c:forEach>

###### 问题5

j-ui 问题为毛毛添加成功之后，关闭当前页面后，列表页面不刷新。

<form action=*"sys/privilege/update?navTabId=sys/privilege/list&callbackType=closeCurrent"* method=*"post"* class=*"pageForm required-validate"* onsubmit=*"return validateCallback(this, navTabAjaxDone);"*>

*navTabId=sys/privilege/list 所对应的值要和初始化rel出来的一样*

*比如：*

<li><a href="/fr-here/rest/sys//privilege/list" target="navTab" rel="sys/privilege/list">查询权限</a></li>

###### 问题6

批量插入时，mybatis不会自动生成id？

问题出现在分配权限时，角色和权限做关联的时候。因为一般都是同时存入一个角色，关联N个权限，所以需要使用批量删除。

最终没有找到解决批量时，id自动生成的问题。

但是发现 ct\_role\_privilege这张表完全可以不使用id，换句话说id就是一个累赘而已。完全没有意思。

所以最后决定去掉id字段。

###### 问题7

**路径问题**

为了避免各种各样的路径问题，尽量采用绝对路径

###### 问题8

参数传入问题

@RequestParam(value = "name",required=false ) int[] names

非必须参数 的写法