版本3 controller方法上设置shiro 注解

Shiro注解作用:

Shiro注解可用在jsp和controller中.用来控制只有拥有注解指定的权限或者角色的用户才能访问相对应的方法或者html代码.

Shiro注解如: @RequiresRoles("r1")

下面是要使用shiro注解需要的配置.

**Spring-mvc.xml**

|  |
| --- |
| **<?**xml *version*="1.0" *encoding*="UTF-8"**?>**  <beans *xmlns*="http://www.springframework.org/schema/beans"  *xmlns:xsi*="http://www.w3.org/2001/XMLSchema-instance" *xmlns:context*="http://www.springframework.org/schema/context"  *xmlns:mvc*="http://www.springframework.org/schema/mvc" *xmlns:aop*="http://www.springframework.org/schema/aop"  *xsi:schemaLocation*="http://www.springframework.org/schema/beans  http://www.springframework.org/schema/beans/spring-beans-4.3.xsd  http://www.springframework.org/schema/context  http://www.springframework.org/schema/context/spring-context-4.3.xsd  http://www.springframework.org/schema/mvc  http://www.springframework.org/schema/mvc/spring-mvc-4.3.xsd  http://www.springframework.org/schema/aop  http://www.springframework.org/schema/aop/spring-aop-4.3.xsd">  *<!-- don't handle the static resource -->*  <mvc:default-servlet-handler />  *<!-- 将指定包下的类，注册到spring -->*  <context:component-scan *base-package*="com.qd.controller" />  *<!-- =======================================使用拦截器防止csrf跨域攻击配置 start ============= -->*  *<!-- 注入给CsrfInterceptor拦截器中的CsrfTokenRepository接口 -->*  <bean *class*="com.qd.csrf.CookieCsrfTokenRepository" />  *<!-- Spring csrf 拦截器 -->*  <mvc:interceptors>  <mvc:interceptor>  <mvc:mapping *path*="/mvc/login" />  <bean *class*="com.qd.interceptor.CsrfInterceptor" />  </mvc:interceptor>  </mvc:interceptors>  *<!-- =========================================使用拦截器防止csrf跨域攻击配置 end ================ -->*  *<!-- 配置结果页面 前缀和后缀 -->*  <bean  *class*="org.springframework.web.servlet.view.InternalResourceViewResolver">  <property *name*="order" *value*="10"></property>  <property *name*="prefix" *value*="/WEB-INF/view/"></property>  <property *name*="suffix" *value*=".jsp"></property>  </bean>  *<!-- 激活mvc注解驱动，就是我们使用@ReqeustMapping这样的注解的时候， 当我们url请求的时候，就可以跟controller中的@requestMapping进行匹配，然后调用对应的方法了。 -->*  <mvc:annotation-driven />  <mvc:annotation-driven>  <mvc:message-converters *register-defaults*="false">  *<!-- 将StringHttpMessageConverter的默认编码设为UTF-8 -->*  <bean *class*="org.springframework.http.converter.StringHttpMessageConverter">  <constructor-arg *value*="UTF-8" />  </bean>  *<!-- json -->*  <ref *bean*="jacksonMessageConverter" />  </mvc:message-converters>  </mvc:annotation-driven>  *<!-- 这个bean，可以使用@ResponseBody的时候，直接返回一个对象，然后spring直接通过这个MessageConverter*  *Bean将对象转换为json。免去了在代码中手动转换的步骤。 -->*  <bean *id*="jacksonMessageConverter"  *class*="org.springframework.http.converter.json.MappingJackson2HttpMessageConverter">  <property *name*="supportedMediaTypes">  <list>  <value>text/html;charset=UTF-8</value>  <value>application/json;charset=UTF-8</value>  </list>  </property>  </bean>  *<!-- 使用spring servlet3.0 默认上传 -->*  *<!-- <bean id="multipartResolver" class="org.springframework.web.multipart.support.StandardServletMultipartResolver"/> -->*  *<!-- <bean id="multipartResolver" class="org.springframework.web.multipart.commons.CommonsMultipartResolver"/> -->*  <bean *id*="multipartResolver"  *class*="org.springframework.web.multipart.commons.CommonsMultipartResolver">  <property *name*="maxUploadSize" *value*="104857600" />  <property *name*="maxInMemorySize" *value*="4096" />  </bean>  *<!--===在方法上使用Shiro的权限相关的注解 ，如@RequiresRoles("USER") 的配置 === start -->*  *<!-- aspectj aop启用CGliB代理 -->*  <aop:aspectj-autoproxy *expose-proxy*="true"  *proxy-target-class*="true" />  *<!-- 要通过shiro的权限来控制是否可以放访问controller中的方法上，就要先声明这个类。 然后就可以用AOP方式，在访问含有@RequiresRoles("USER")*  *相关shiro注解的方法的时候， 先让shiro去判断当前用户是否有对应的权限。有权限才能调用此方法。 与上面aop:aspectj合用才行 -->*  <bean  *class*="org.apache.shiro.spring.security.interceptor.AuthorizationAttributeSourceAdvisor">  <property *name*="securityManager" *ref*="securityManager" />  </bean>  *<!-- ==在方法上使用Shiro的权限相关的注解 ，如@RequiresRoles("USER") 的配置 ===end-->*  </beans> |

上面绿色边框部分,就是要使用注解需要添加的配置.

下面是shiro注解的使用:

LoginControler.java中的方法。详细请看项目中的代码



登录成功后,可以通过url直接访问此路径,来测试这个权限注解的使用