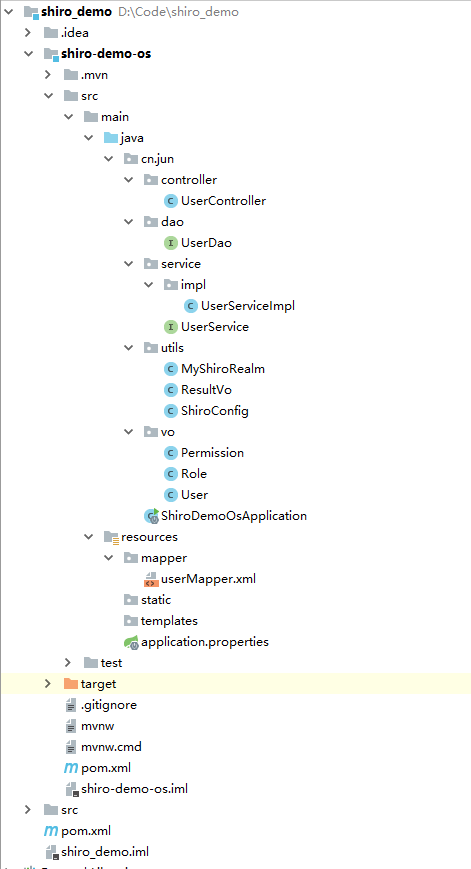
1. 目录结构



2. 编码流程

（1）新建一个maven项目

（2）在项目中新建一个后台module

（3）首先编写一个正常ssm的dao、service、controller

（4）编写shiro部分

（5）修改controller

（6）postman测试

3. 代码详情

pom.xml:

<**dependencies**>  
 <**dependency**>  
 <**groupId**>org.springframework.boot</**groupId**>  
 <**artifactId**>spring-boot-starter-data-redis</**artifactId**>  
 </**dependency**>  
 <**dependency**>  
 <**groupId**>org.springframework.boot</**groupId**>  
 <**artifactId**>spring-boot-starter-jdbc</**artifactId**>  
 </**dependency**>  
 <**dependency**>  
 <**groupId**>org.springframework.boot</**groupId**>  
 <**artifactId**>spring-boot-starter-web</**artifactId**>  
 </**dependency**>  
 <**dependency**>  
 <**groupId**>org.mybatis.spring.boot</**groupId**>  
 <**artifactId**>mybatis-spring-boot-starter</**artifactId**>  
 <**version**>1.3.2</**version**>  
 </**dependency**>  
 <**dependency**>  
 <**groupId**>mysql</**groupId**>  
 <**artifactId**>mysql-connector-java</**artifactId**>  
 <**scope**>runtime</**scope**>  
 </**dependency**>  
 <**dependency**>  
 <**groupId**>org.springframework.boot</**groupId**>  
 <**artifactId**>spring-boot-starter-test</**artifactId**>  
 <**scope**>test</**scope**>  
 </**dependency**>  
 <**dependency**>  
 <**groupId**>org.apache.shiro</**groupId**>  
 <**artifactId**>shiro-spring-boot-starter</**artifactId**>  
 <**version**>1.4.0</**version**>  
 </**dependency**>  
</**dependencies**>

springboot配置文件：

**spring.datasource.url** = **jdbc:mysql://localhost:3306/login\_test?serverTimezone=GMT%2B8  
spring.datasource.username** = **root  
spring.datasource.password** = **root  
spring.datasource.driverClassName** = **com.mysql.cj.jdbc.Driver  
server.port**=**80***#加载mybatis对应的mapper位置***mybatis.mapper-locations**=**classpath:mapper/\*.xml***#加载mybatis别名设置***mybatis.type-aliases-package**=**cn.jun.vo***# 驼峰命名规范 如：数据库字段是 order\_id 那么 实体字段就要写成 orderId***mybatis.configuration.map-underscore-to-camel-case**=**true**

shiro部分：

**public class** MyShiroRealm **extends** AuthorizingRealm {  
  
 @Autowired  
 **private** UserDao **userDao**;  
  
  
 *//角色权限和对应权限添加* @Override  
 **protected** AuthorizationInfo doGetAuthorizationInfo(PrincipalCollection principalCollection) {  
 System.***out***.println(**"doGetAuthorizationInfo:角色权限和对应权限添加"**);  
 *//获取登陆用户名* User loginUser = (User)principalCollection.getPrimaryPrincipal();  
 *//添加权限和角色* SimpleAuthorizationInfo info = **new** SimpleAuthorizationInfo();  
 loginUser.getRoles().forEach(role -> {  
 *//添加角色* info.addRole(role.getRoleName());  
 *//添加权限* role.getPermissions().forEach(permission -> {  
 info.addStringPermission(permission.getPermissionName());  
 });  
 });  
 **return** info;  
 }  
  
 *//用户认证* @Override  
 **protected** AuthenticationInfo doGetAuthenticationInfo(AuthenticationToken authenticationToken) **throws** AuthenticationException {  
 System.***out***.println(**"doGetAuthenticationInfo:用户认证"**);  
 *//加这一步的目的是在Post请求的时候会先进认证，然后在到请求* **if** (authenticationToken.getPrincipal() == **null**) {  
 **return null**;  
 }  
 *//获取用户信息* UsernamePasswordToken token = (UsernamePasswordToken)authenticationToken;  
 User user = **userDao**.getUserByName(token.getUsername());  
 **if**(user== **null**){  
 **return null**;  
 }  
 **return new** SimpleAuthenticationInfo(user, user.getPassword(), getName());  
 }  
}

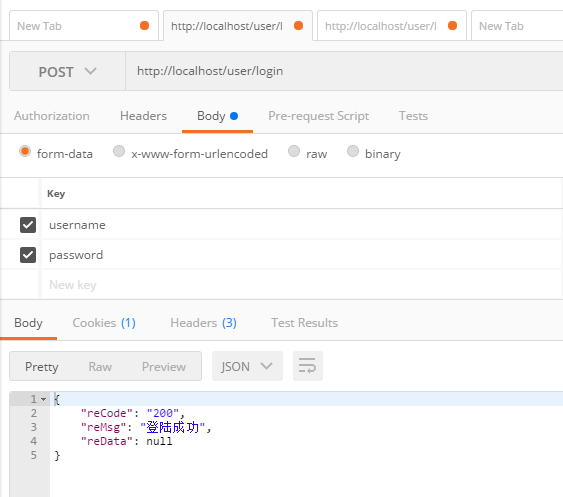
@Configuration  
**public class** ShiroConfig {  
  
 *//将自己的验证方式加入容器* @Bean  
 **public** MyShiroRealm myShiroRealm(){  
 MyShiroRealm myShiroRealm = **new** MyShiroRealm();  
 **return** myShiroRealm;  
 }  
  
 *//权限管理，配置Realm的管理认证* @Bean  
 **public** DefaultWebSecurityManager securityManager(){  
 DefaultWebSecurityManager securityManager = **new** DefaultWebSecurityManager();  
 securityManager.setRealm(myShiroRealm());  
 **return** securityManager;  
 }  
  
 *//Filter工厂，设置对应的过滤条件和跳转条件* @Bean  
 **public** ShiroFilterFactoryBean shiroFilterFactoryBean(SecurityManager securityManager){  
 ShiroFilterFactoryBean shiroFilterFactoryBean = **new** ShiroFilterFactoryBean();  
 shiroFilterFactoryBean.setSecurityManager(securityManager);  
 Map<String,String> map = **new** HashMap<>();  
 *//登出* map.put(**"/logout"**,**"logout"**);  
 *//对所有用户认证* map.put(**"/\*\*"**,**"authc"**);  
 *//登陆* shiroFilterFactoryBean.setLoginUrl(**"/user/login"**);  
 *//登陆后首页* shiroFilterFactoryBean.setSuccessUrl(**"/user/index"**);  
 *//登陆错误页* shiroFilterFactoryBean.setUnauthorizedUrl(**"/error"**);  
 shiroFilterFactoryBean.setFilterChainDefinitionMap(map);  
 **return** shiroFilterFactoryBean;  
 }  
  
 *//加入注解的使用* @Bean  
 **public** AuthorizationAttributeSourceAdvisor authorizationAttributeSourceAdvisor(SecurityManager securityManager){  
 AuthorizationAttributeSourceAdvisor authorizationAttributeSourceAdvisor = **new** AuthorizationAttributeSourceAdvisor();  
 authorizationAttributeSourceAdvisor.setSecurityManager(securityManager);  
 **return** authorizationAttributeSourceAdvisor;  
 }  
  
}

controller部分：

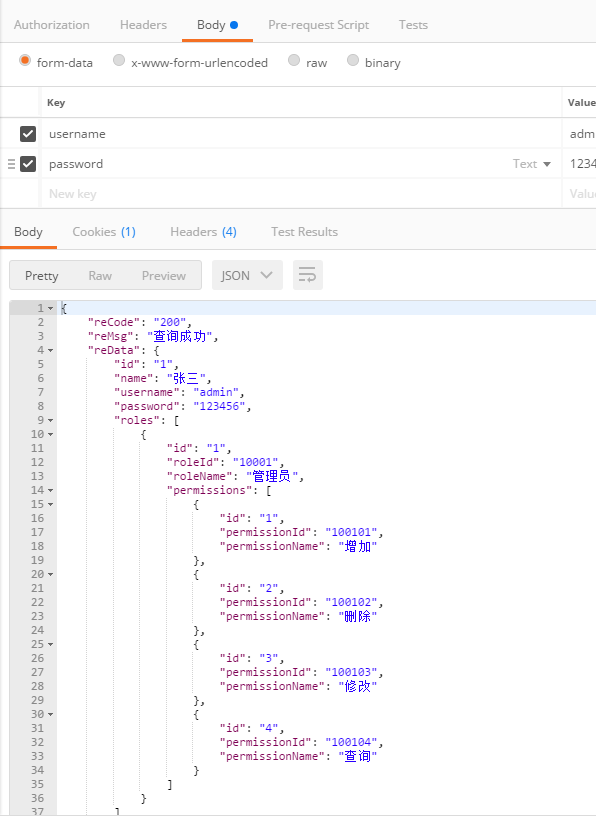
@Controller  
@RequestMapping(**"/user"**)  
**public class** UserController {  
  
 @Autowired  
 **private** UserService **userService**;  
  
 @RequestMapping(value = **"/login"**,method = RequestMethod.***POST***)  
 @ResponseBody  
 **public** ResultVo login(User user){  
 Subject subject = SecurityUtils.*getSubject*();  
 UsernamePasswordToken usernamePasswordToken = **new** UsernamePasswordToken(user.getUsername(),user.getPassword());  
 subject.login(usernamePasswordToken);  
 **return userService**.login(user);  
 }  
  
 @RequestMapping(value = **"/index"**)  
 @ResponseBody  
 **public** ResultVo index(){  
 ResultVo resultVo = **new** ResultVo(**"200"**,**"登陆成功"**);  
 **return** resultVo;  
 }  
  
 *//登出* @RequestMapping(value = **"/logout"**)  
 @ResponseBody  
 **public** String logout(){  
 **return "logout"**;  
 }  
  
 *//错误页面展示* @RequestMapping(value = **"/error"**,method = RequestMethod.***POST***)  
 **public** String error(){  
 **return "error ok!"**;  
  
 }  
  
 *//注解的使用* @RequiresRoles(**"admin"**)  
 @RequiresPermissions(**"create"**)  
 @RequestMapping(value = **"/create"**)  
 **public** String create(){  
 **return "Create success!"**;  
 }  
}

4. 结果验证

第一次请求：



第二次请求：



5. github地址：

https://github.com/1064399445/shiro-demo.git