# 第三章：shiro集成spring

## ssm环境搭建

### 添加jar依赖

|  |
| --- |
| <!-- 添加Servlet支持 -->  <dependency>  <groupId>javax.servlet</groupId>  <artifactId>javax.servlet-api</artifactId>  <version>3.1.0</version>  </dependency>  <dependency>  <groupId>javax.servlet.jsp</groupId>  <artifactId>javax.servlet.jsp-api</artifactId>  <version>2.3.1</version>  </dependency>  <!-- 添加jtl支持 -->  <dependency>  <groupId>javax.servlet</groupId>  <artifactId>jstl</artifactId>  <version>1.2</version>  </dependency>  <!-- 添加Spring支持 -->  <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-core</artifactId>  <version>4.1.7.RELEASE</version>  </dependency>  <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-beans</artifactId>  <version>4.1.7.RELEASE</version>  </dependency>  <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-tx</artifactId>  <version>4.1.7.RELEASE</version>  </dependency>  <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-context</artifactId>  <version>4.1.7.RELEASE</version>  </dependency>  <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-context-support</artifactId>  <version>4.1.7.RELEASE</version>  </dependency>  <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-web</artifactId>  <version>4.1.7.RELEASE</version>  </dependency>  <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-webmvc</artifactId>  <version>4.1.7.RELEASE</version>  </dependency>  <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-aop</artifactId>  <version>4.1.7.RELEASE</version>  </dependency>  <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-aspects</artifactId>  <version>4.1.7.RELEASE</version>  </dependency>  <dependency>  <groupId>org.springframework</groupId>  <artifactId>spring-jdbc</artifactId>  <version>4.1.7.RELEASE</version>  </dependency>  <dependency>  <groupId>org.mybatis</groupId>  <artifactId>mybatis-spring</artifactId>  <version>1.2.3</version>  </dependency>  <!-- 添加日志支持 -->  <dependency>  <groupId>log4j</groupId>  <artifactId>log4j</artifactId>  <version>1.2.17</version>  </dependency>  <!-- 添加mybatis支持 -->  <dependency>  <groupId>org.mybatis</groupId>  <artifactId>mybatis</artifactId>  <version>3.3.0</version>  </dependency>  <!-- jdbc驱动包 -->  <dependency>  <groupId>mysql</groupId>  <artifactId>mysql-connector-java</artifactId>  <version>8.0.15</version>  </dependency>  <dependency>  <groupId>org.apache.shiro</groupId>  <artifactId>shiro-core</artifactId>  <version>1.2.4</version>  </dependency>  <dependency>  <groupId>org.slf4j</groupId>  <artifactId>slf4j-log4j12</artifactId>  <version>1.7.12</version>  </dependency>  <dependency>  <groupId>org.apache.shiro</groupId>  <artifactId>shiro-web</artifactId>  <version>1.2.4</version>  </dependency>  <dependency>  <groupId>org.apache.shiro</groupId>  <artifactId>shiro-spring</artifactId>  <version>1.2.4</version>  </dependency> |

### 添加spring配置文件

|  |
| --- |
| <!-- 自动扫描 -->  <context:component-scan base-package=*"com.shiro.service"* />  <!-- 配置数据源 -->  <bean id=*"dataSource"*  class=*"org.springframework.jdbc.datasource.DriverManagerDataSource"*>  <property name=*"driverClassName"* value=*"com.mysql.cj.jdbc.Driver"*/>  <property name=*"url"* value=*"jdbc:mysql://localhost:3306/db\_shiro?useUnicode=true&amp;characterEncoding=utf-8&amp;serverTimezone=UTC"*/>  <property name=*"username"* value=*"root"*/>  <property name=*"password"* value=*"root"*/>  </bean>  <!-- 配置mybatis的sqlSessionFactory -->  <bean id=*"sqlSessionFactory"* class=*"org.mybatis.spring.SqlSessionFactoryBean"*>  <property name=*"dataSource"* ref=*"dataSource"* />  <!-- mybatis配置文件 -->  <property name=*"configLocation"* value=*"classpath:mybatis-config.xml"*></property>  </bean>  <!-- DAO接口所在包名，Spring会自动查找其下的类 -->  <bean class=*"org.mybatis.spring.mapper.MapperScannerConfigurer"*>  <property name=*"basePackage"* value=*"com.shiro.dao"* />  <property name=*"sqlSessionFactoryBeanName"* value=*"sqlSessionFactory"*></property>  </bean>  <!-- (事务管理)transaction manager, use JtaTransactionManager for global tx -->  <bean id=*"transactionManager"*  class=*"org.springframework.jdbc.datasource.DataSourceTransactionManager"*>  <property name=*"dataSource"* ref=*"dataSource"* />  </bean>  <!-- 配置事务通知属性 -->  <tx:advice id=*"txAdvice"* transaction-manager=*"transactionManager"*>  <!-- 定义事务传播属性 -->  <tx:attributes>  <tx:method name=*"insert\*"* propagation=*"REQUIRED"* />  <tx:method name=*"update\*"* propagation=*"REQUIRED"* />  <tx:method name=*"edit\*"* propagation=*"REQUIRED"* />  <tx:method name=*"save\*"* propagation=*"REQUIRED"* />  <tx:method name=*"add\*"* propagation=*"REQUIRED"* />  <tx:method name=*"new\*"* propagation=*"REQUIRED"* />  <tx:method name=*"set\*"* propagation=*"REQUIRED"* />  <tx:method name=*"remove\*"* propagation=*"REQUIRED"* />  <tx:method name=*"delete\*"* propagation=*"REQUIRED"* />  <tx:method name=*"change\*"* propagation=*"REQUIRED"* />  <tx:method name=*"check\*"* propagation=*"REQUIRED"* />  <tx:method name=*"get\*"* propagation=*"REQUIRED"* read-only=*"true"* />  <tx:method name=*"find\*"* propagation=*"REQUIRED"* read-only=*"true"* />  <tx:method name=*"load\*"* propagation=*"REQUIRED"* read-only=*"true"* />  <tx:method name=*"\*"* propagation=*"REQUIRED"* read-only=*"true"* />  </tx:attributes>  </tx:advice>    <!-- 配置事务切面 -->  <aop:config>  <aop:pointcut id=*"serviceOperation"*  expression=*"execution(\* com.shiro.service.\*.\*(..))"* />  <aop:advisor advice-ref=*"txAdvice"* pointcut-ref=*"serviceOperation"* />  </aop:config> |

### 添加mybatis配置文件

|  |
| --- |
| <?xml version=*"1.0"* encoding=*"UTF-8"* ?>  <!DOCTYPE configuration  PUBLIC "-//mybatis.org//DTD Config 3.0//EN"  "http://mybatis.org/dtd/mybatis-3-config.dtd">  <configuration>  <!-- 别名 -->  <typeAliases>  <package name=*"com.shiro.entity"*/>  </typeAliases>  </configuration> |

### springmvc配置文件

|  |
| --- |
| <!-- 扫描注解 -->  <context:component-scan base-package=*"com.shiro.controller"*/>    <!-- 注入驱动 -->  <mvc:annotation-driven/>    <!-- 视图解析器 -->  <bean id=*"viewResolver"*  class=*"org.springframework.web.servlet.view.InternalResourceViewResolver"*>  <property name=*"prefix"* value=*"/"* />  <property name=*"suffix"* value=*".jsp"*></property>  </bean> |

### web.xml配置文件

|  |
| --- |
| <!-- Spring配置文件 -->  <context-param>  <param-name>contextConfigLocation</param-name>  <param-value>classpath:applicationContext.xml</param-value>  </context-param>  <!-- 编码过滤器 -->  <filter>  <filter-name>encodingFilter</filter-name>  <filter-class>org.springframework.web.filter.CharacterEncodingFilter</filter-class>  <async-supported>true</async-supported>  <init-param>  <param-name>encoding</param-name>  <param-value>UTF-8</param-value>  </init-param>  </filter>  <filter-mapping>  <filter-name>encodingFilter</filter-name>  <url-pattern>/\*</url-pattern>  </filter-mapping>  <!-- Spring监听器 -->  <listener>  <listener-class>org.springframework.web.context.ContextLoaderListener</listener-class>  </listener>  <!-- 添加对springmvc的支持 -->  <servlet>  <servlet-name>springMVC</servlet-name>  <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>  <init-param>  <param-name>contextConfigLocation</param-name>  <param-value>classpath:springmvc.xml</param-value>  </init-param>  <load-on-startup>1</load-on-startup>  <async-supported>true</async-supported>  </servlet>  <servlet-mapping>  <servlet-name>springMVC</servlet-name>  <url-pattern>\*.action</url-pattern>  </servlet-mapping> |

## ssm+shiro实现登录

### 数据库设计

|  |
| --- |
|  |

### 实体类

|  |
| --- |
| **public** **class** User {  **private** Integer id;  **private** String userName;  **private** String password;  **public** Integer getId() {  **return** id;  }  **public** **void** setId(Integer id) {  **this**.id = id;  }  **public** String getUserName() {  **return** userName;  }  **public** **void** setUserName(String userName) {  **this**.userName = userName;  }  **public** String getPassword() {  **return** password;  }  **public** **void** setPassword(String password) {  **this**.password = password;  }  @Override  **public** String toString() {  **return** "User [id=" + id + ", userName=" + userName + ", password=" + password + "]";  }  } |

### mapper接口

|  |
| --- |
|  |

### mapper映射文件

|  |
| --- |
|  |

### service接口

|  |
| --- |
|  |

### service实现类

|  |
| --- |
|  |

### controller控制器

|  |
| --- |
|  |

### login页面

|  |
| --- |
|  |

## 自定义realm之身份验证

|  |
| --- |
|  |

## shiro集成spring

### spring配置文件

|  |
| --- |
| <!-- 1.注入自定义realm -->  <bean id=*"myRealm"* class=*"com.shiro.realm.UserRealm"* />  <!-- 2.将realm注入到securityManager安全管理器 -->  <bean id=*"securityManager"*  class=*"org.apache.shiro.web.mgt.DefaultWebSecurityManager"*>  <property name=*"realm"* ref=*"myRealm"* />  </bean>  <!-- 3.Shiro过滤器 -->  <bean id=*"shiroFilter"*  class=*"org.apache.shiro.spring.web.ShiroFilterFactoryBean"*>  <!-- Shiro的核心安全接口,这个属性是必须的 -->  <property name=*"securityManager"* ref=*"securityManager"* />  <!-- 身份认证失败，则跳转到登录页面的配置 -->  <property name=*"loginUrl"* value=*"/login.jsp"* />  <!-- 权限认证失败，则跳转到指定页面 -->  <property name=*"unauthorizedUrl"* value=*"/unauthorized.jsp"* />  <!-- Shiro连接约束配置,即过滤链的定义 -->  <property name=*"filterChainDefinitions"*>  <value>  <!-- 登录请求可匿名访问 -->  /user/login.action = anon  <!-- admin下所有请求必须进行身份验证且用户必须拥有admin角色 -->  /admin/\*\* = authc,roles[admin]  <!-- student请求必须拥有teacher角色 -->  /student = roles[teacher]  <!-- teacher请求必须拥有创建用户的权限 -->  /teacher = perms["user:create"]  <!-- 以下所有请求必须经过身份验证才可访问 -->  /\*\* = authc  </value>  </property>  </bean>  <!-- 4.保证实现了Shiro内部lifecycle函数的bean执行 -->  <bean id=*"lifecycleBeanPostProcessor"*  class=*"org.apache.shiro.spring.LifecycleBeanPostProcessor"* />  <!-- 5.开启Shiro注解 -->  <bean  class=*"org.springframework.aop.framework.autoproxy.DefaultAdvisorAutoProxyCreator"*  depends-on=*"lifecycleBeanPostProcessor"* />  <bean  class=*"org.apache.shiro.spring.security.interceptor.AuthorizationAttributeSourceAdvisor"*>  <property name=*"securityManager"* ref=*"securityManager"* />  </bean> |

### web.xml配置文件

|  |
| --- |
| <!-- shiro配置 -->  <!-- shiro过滤器定义 -->  <filter>  <filter-name>shiroFilter</filter-name>  <filter-class>org.springframework.web.filter.DelegatingFilterProxy</filter-class>  <init-param>  <!-- 该值缺省为false,表示生命周期由SpringApplicationContext管理,设置为true则表示由ServletContainer管理 -->  <param-name>targetFilterLifecycle</param-name>  <param-value>true</param-value>  </init-param>  </filter>  <filter-mapping>  <filter-name>shiroFilter</filter-name>  <url-pattern>/\*</url-pattern>  </filter-mapping> |

提示：完成以上代码及配置，即可测试登录功能

## ssm + shiro实现权限验证

### mapper接口

|  |
| --- |
|  |

### mapper映射文件

|  |
| --- |
|  |

### service接口

|  |
| --- |
|  |

### service实现类

|  |
| --- |
|  |

## 自定义realm之授权

|  |
| --- |
|  |

完成以上编码配置， 即可测试授权

测试方法：

1. 登录拥有admin角色的用户
   1. 访问/admin请求【正确效果：404错误】
   2. 访问/teacher请求【正确效果：404错误】
   3. 访问/student请求【正确效果：去到权限不足页面】
2. 其他角色测试方法类似于上面的测试方法