Shiro 在项目中实施安全控制，权限分配的实战应用

1. shiro 与 我们手写的rbac 的区别与联系

不管是shiro ,还是 rbac ,其目的都是为我们的项目，实施安全验证与权限管理，它们涉及到的数据表也是一样的： admin 表，role 表, resources表，

Shiro 其实是对rbac 框架的一个高度的封装，使得我们能更便捷地使用rbac功能，它提供一系列的API (类和接口),简化安全与权限控制，比如，它提供了一个叫realm 的类，我们实现它，就可以在realm的实例 对象中编写安全与权限代码，它提供现成的annotation,方便我们控制controller 及 action 的访问控制，它提供shiro 标签 ，方便我们控制页面元素的访问权限

Shiro 在web 项目中， 是以一个前端控制器（过滤器）开始发生作用。

Shiro : 四大功能：Authentication 认证器

Authorization 权限器

Session management 会话管理

Cryptgraphy 加密

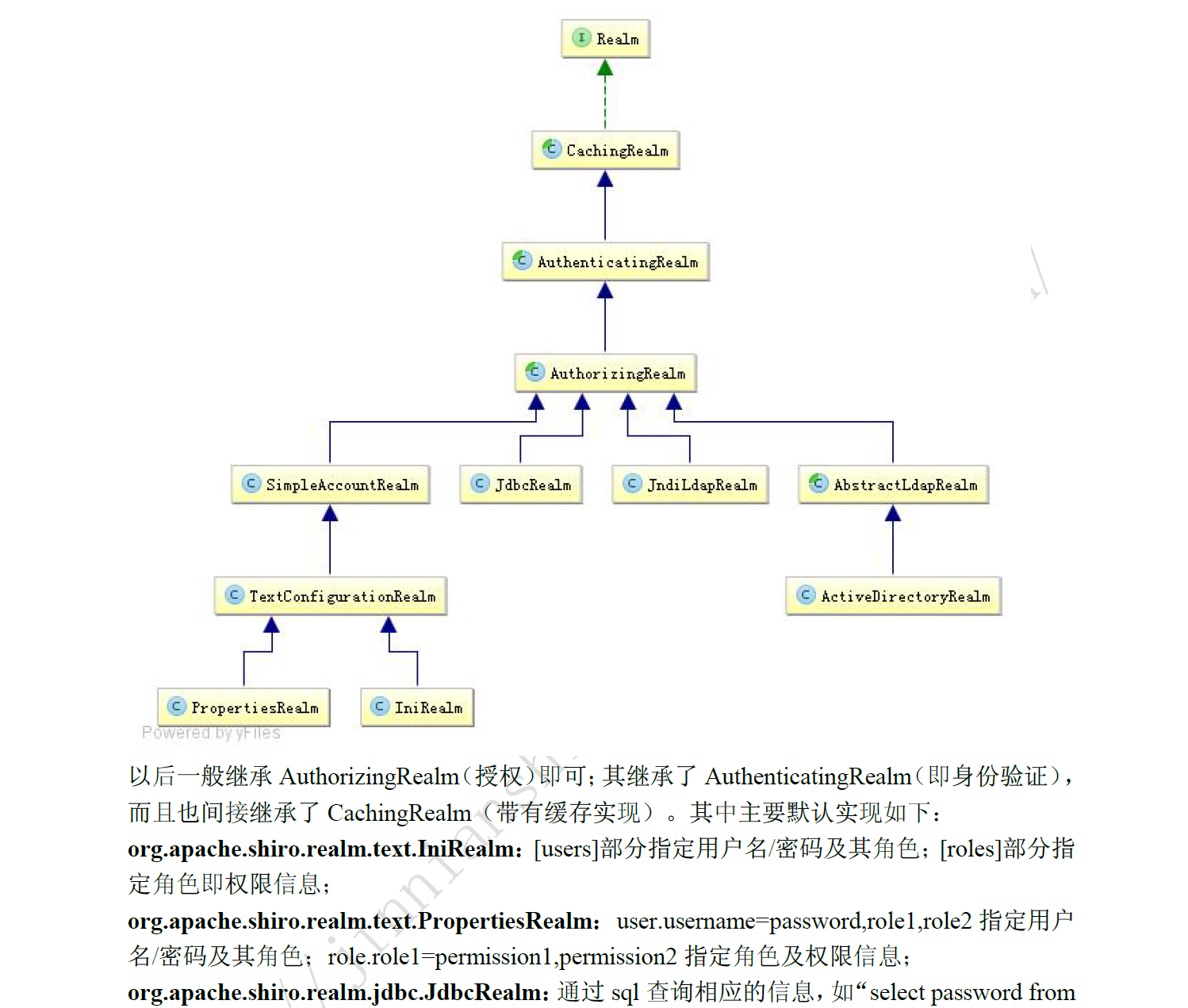
**记住一点，Shiro 不会去维护用户、维护权限；这些需要我们自己去设计/提供；然后通过相应的接口注入给Shiro即可。**

Shiro 在项目实战中的应用步骤：

1. 导包（在maven 当中就是贴坐标）
2. 在 web.xml 配置文件中，配置shiro 过滤器------- 配置spring 的 delegatingFilterProxy 将过滤代理委托给一个shiro 类，这个 委托代理类就是 shiroFilter,要在spring 的上下文配置中详细配置
3. 编写shiro 配置文件(整合到spring 的配置文件applicationContext.xml 文件中进行配置

|  |
| --- |
| <bean id="credentialsMatcher" class="org.apache.shiro.authc.credential.HashedCredentialsMatcher">  <property name="hashAlgorithmName" value="md5"/>  <property name="hashIterations" value="2"/>  </bean> |
| **告知shiro ------ 密码认证，以name 为盐，做两次md5加密，来密码验证** |
| <!-- 自定义Realm -->  <bean id="myRealm" class="org.consiste.util.realm.MyRealm">  <property name="credentialsMatcher" ref="credentialsMatcher"></property>    </bean> |
| **Realm 是shiro 的核心组件，我们在其中为它指明密码验证策略** |
| <!-- 安全管理器 -->  <bean id="securityManager" class="org.apache.shiro.web.mgt.DefaultWebSecurityManager">  <property name="realm" ref="myRealm" />  </bean> |
| **我们只能调用安全器，由安全器去调用realm 的功能** |
| <!-- 相当于调用SecurityUtils.setSecurityManager(securityManager) -->  <bean class="org.springframework.beans.factory.config.MethodInvokingFactoryBean">  <property name="staticMethod" value="org.apache.shiro.SecurityUtils.setSecurityManager"/>  <property name="arguments" ref="securityManager"/>  </bean> |
| **为静态方法调用指定方法调用工厂bean ------ 为 SecurityUtil指定安全管理器** |
| **<!-- 基于Form表单的身份验证过滤器 -->**  **<bean id="formAuthenticationFilter" class="org.apache.shiro.web.filter.authc.FormAuthenticationFilter">**  **<property name="usernameParam" value="uname"/>**  **<property name="passwordParam" value="password"/>**  **<property name="rememberMeParam" value="rememberMe"/>**  **<property name="loginUrl" value="/login"/>**  **</bean>** |
| **指定表单验证页面----登录页面，注意:那个value 的值要和页面上 input 的name 一致** |
| **<bean id="logout" class="org.apache.shiro.web.filter.authc.LogoutFilter">**  **<property name="redirectUrl" value="/login"/>**  **</bean>** |
| **Logout 的配置，其中的property 指明：logout 以后，去哪儿？此处，我们是去登录页面** |
| **<!-- 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" />**  **<!-- 登录成功后跳转路径 -->**  **<property name="successUrl" value="/index" />**    **<!-- 权限认证失败，则跳转到指定页面 -->**  **<property name="unauthorizedUrl" value="/common/unauthor.jsp" />**  **<!-- Shiro连接约束配置,即过滤链的定义 -->**  **<property name="filterChainDefinitions">**  **<value>**      **/kaptcha.jpg=anon**  **/validator\_code\_check=anon**  **/resources/\*\*=anon**    **/login=authc**  **/logout=logout**      **/\*\*=authc**  **<!--**  **/survy/\*\*=authc**  **/sys/\*\*=authc**  **-->**  **</value>**  **</property>**  **</bean>** |
| **配置shiro 过滤器** |
| **<!-- 保证实现了Shiro内部lifecycle函数的bean执行 -->**  **<bean id="lifecycleBeanPostProcessor" class="org.apache.shiro.spring.LifecycleBeanPostProcessor" />** |
| **<!-- 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>** |
|  |
|  |
|  |
|  |

**3-1** : 先自定义一个realm ,然后在 applicationContext.xml 文件中声明



Shiro 的相关名词解释

1. Subject ---- 主体：要去使用这个应用程 序的人或软件: 通常我们会将Subject对象理解为一个用户，同样的它也有可能是一个三方程序，它是一个抽象的概念，可以理解为任何与系统交互的“东西”都是Subject。
2. Realm ------ 是shiro 的核心，也可以理解为安全数据源，验证信息要从realma 中得到，授权信息(permission)， 我们也要从realm 中得到------- 而这些安全和授权信息，并非天生就有， 是我们手工在realm 的代码中设定的

3 principal ----- 字面意思，应该 是：身份，它可以是可以是uuid  数据库中的主键  LDAP UUID或静态DN  在所有用户帐户中唯一的字符串用户名。

4 credential ------ 证书 赁证 的意思

5 token ----- 令牌： 其中包含了一个身份以及对应的凭证：是一个 身份：凭证 对，象键值对一样