# Shiro

## apache shiro框架简介

官网：shiro.apache.org



* 下载文件：



* shiro框架的核心功能：

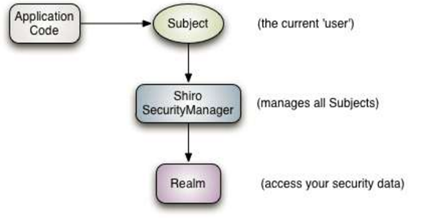
认证

授权

会话管理

加密

* shiro框架认证流程

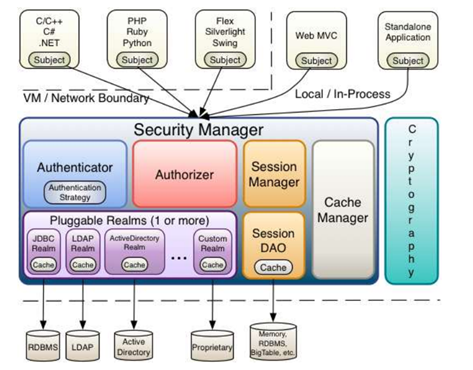


Application Code：应用程序代码，由开发人员负责开发的

Subject：框架提供的接口，代表当前用户对象

SecurityManager：框架提供的接口，代表安全管理器对象

Realm：可以开发人员编写，框架也提供一些，类似于ＤＡＯ，用于访问权限数据

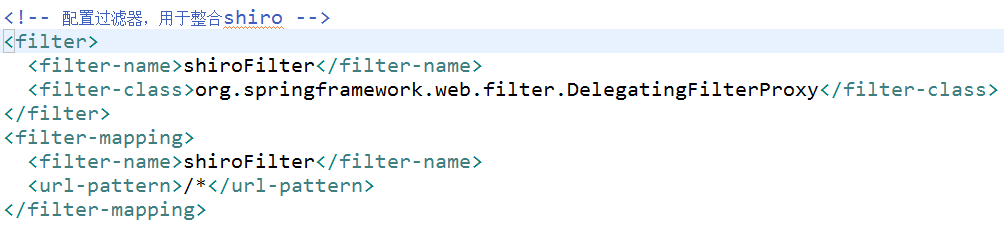


## 1.项目中使用框架步骤

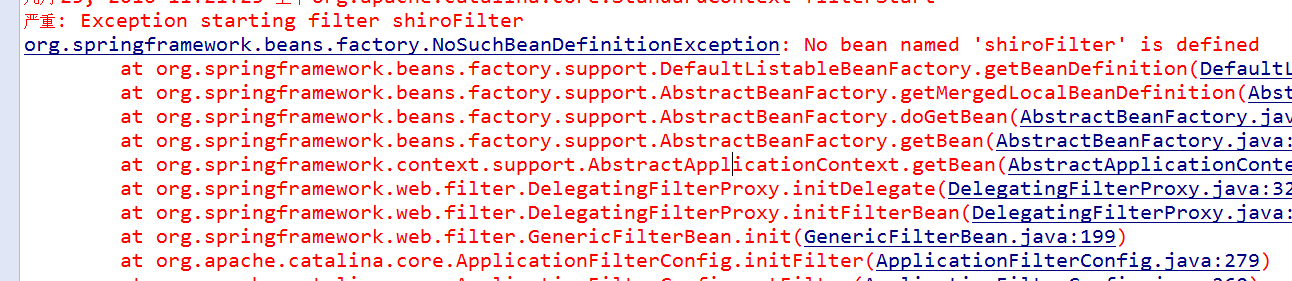
第一步：引入shiro框架相关的jar



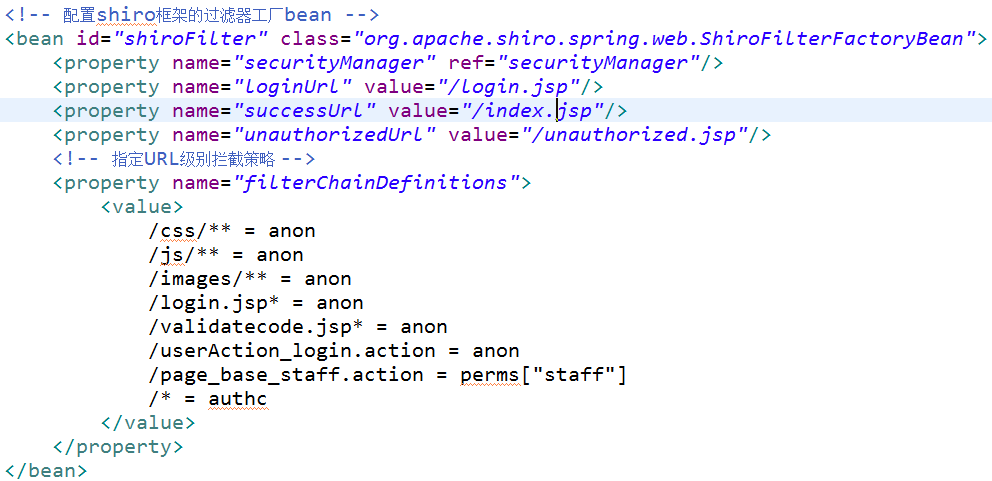
第二步：在web.xml中配置spring框架提供的用于整合shiro框架的过滤器



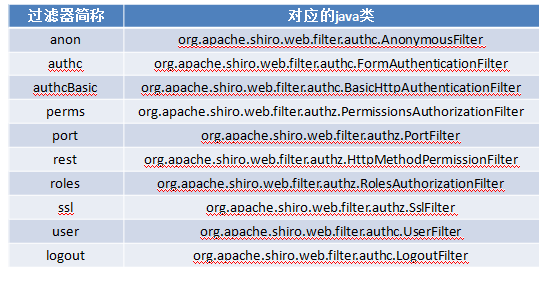
启动tomcat服务器，抛出异常：spring工厂中不存在一个名称为“shiroFilter”的bean对象



第三步：在spring配置文件中配置bean，id为shiroFilter



框架提供的过滤器：



第四步：配置安全管理器



第五步：修改UserAction中的login方法，使用shiro提供的方式进行认证操作



第六步：自定义realm，并注入给安全管理器

**public** **class** BOSRealm **extends** AuthorizingRealm{

@Autowired

**private** IUserDao userDao;

//认证方法

**protected** AuthenticationInfo doGetAuthenticationInfo(AuthenticationToken token) **throws** AuthenticationException {

System.***out***.println("realm中的认证方法执行了。。。。");

UsernamePasswordToken mytoken = (UsernamePasswordToken)token;

String username = mytoken.getUsername();

//根据用户名查询数据库中的密码

User user = userDao.findUserByUserName(username);

**if**(user == **null**){

//用户名不存在

**return** **null**;

}

//如果能查询到，再由框架比对数据库中查询到的密码和页面提交的密码是否一致

AuthenticationInfo info = **new** SimpleAuthenticationInfo(user, user.getPassword(), **this**.getName());

**return** info;

}

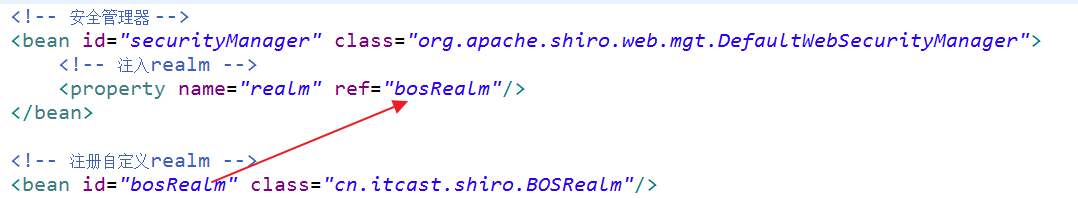
//授权方法

**protected** AuthorizationInfo doGetAuthorizationInfo(PrincipalCollection principals) {

// **TODO** Auto-generated method stub

**return** **null**;

}



## 2.在realm中进行授权

在BOSRealm中实现授权方法：



## 3.使用shiro的方法注解方式权限控制

第一步：在spring配置文件中开启shiro注解支持

<!-- 开启shiro框架注解支持 -->

<bean id=*"defaultAdvisorAutoProxyCreator"*

class=*"org.springframework.aop.framework.autoproxy.DefaultAdvisorAutoProxyCreator"*>

<!-- 必须使用cglib方式为Action对象创建代理对象 -->

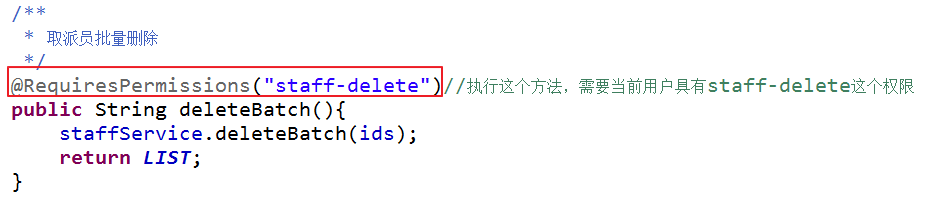
<property name=*"proxyTargetClass"* value=*"true"*/>

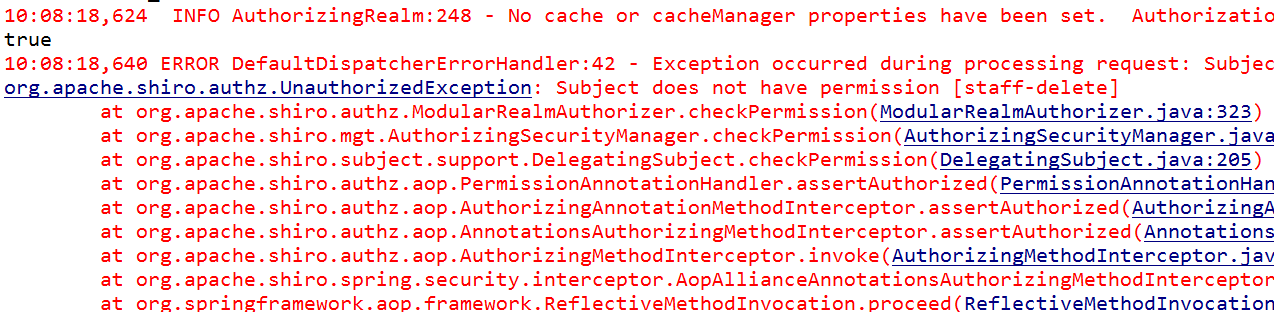
</bean>

<!-- 配置shiro框架提供的切面类，用于创建代理对象 -->

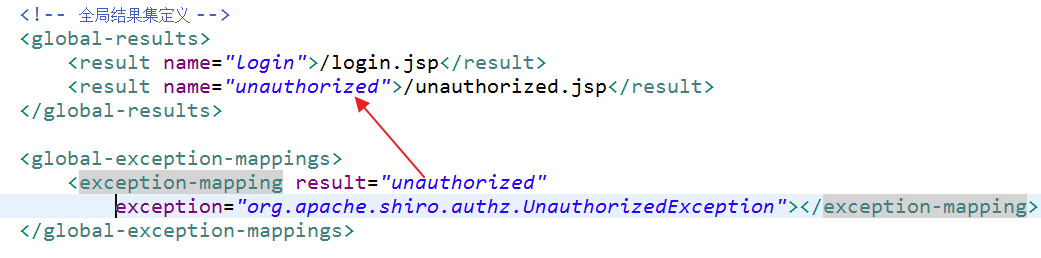
<bean class=*"org.apache.shiro.spring.security.interceptor.AuthorizationAttributeSourceAdvisor"*/>

第二步：在Action的方法上使用shiro注解





第三步：在struts.xml中配置全局异常捕获，当shiro框架抛出权限不足异常时，跳转到权限不足提示页面

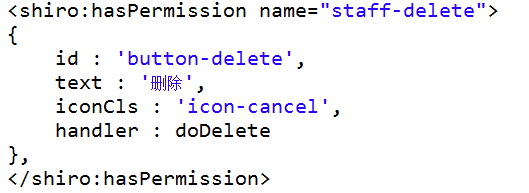


## 4.使用shiro提供的页面标签方式权限控制

第一步：在jsp页面中引入shiro的标签库

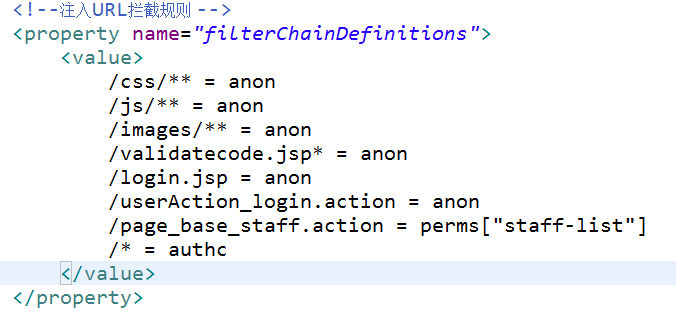
<%@ taglib prefix="shiro" uri="http://shiro.apache.org/tags" %>

第二步：使用shiro的标签控制页面元素展示

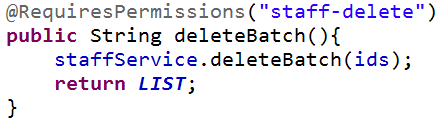


## 5.总结shiro框架提供的权限控制方式

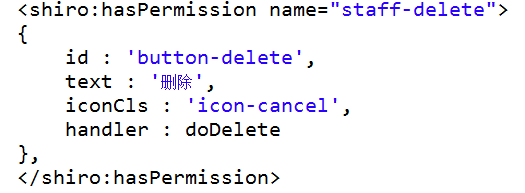
* URL拦截权限控制（基于过滤器实现）



* 方法注解权限控制（基于代理技术实现）



* 页面标签权限控制（标签技术实现）



* 代码级别权限控制（基于代理技术实现）



# 使用ehcache缓存权限数据

ehcache是专门缓存插件，可以缓存Java对象，提高系统性能。

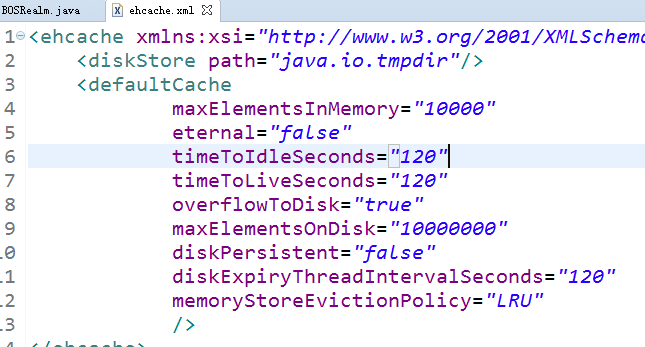
* ehcache提供的jar包：



第一步：在pom.xml文件中引入ehcache的依赖



第二步：在项目中提供ehcache的配置文件



第三步：在spring配置文件中配置缓存管理器对象，并注入给安全管理器对象

