# 统一鉴权

## 获取access\_token

*/\*\*  
 \* 提供登录接口，授予token  
 \** ***@param*** *username  
 \** ***@param*** *password  
 \** ***@return*** *\** ***@throws*** *AuthzException 用户名或密码错误将抛出异常  
 \*/*Token getToken(String username, String password) throws AuthzException;

**实现如下：**



## 鉴权

*/\*\*  
 \* 判断是否有权限  
 \** ***@param*** *accessToken  
 \** ***@param*** *permission  
 \** ***@throws*** *AuthzException 没有权限抛出异常  
 \*/*void checkPermission(String accessToken, String permission) throws AuthzException;

**实现如下：**



**此外还提供另一个接口（对上面接口的简单包装）：**

*/\*\*  
 \* 判断是否有权限  
 \** ***@param*** *accessToken  
 \** ***@param*** *permission  
 \** ***@return*** *当且仅当有权限返回true  
 \*/*boolean isPermitted(String accessToken, String permission);

## 额外话

“统一鉴权”只使用了shiro的授权功能。至于授权接口，定义在Authorizer。调用该接口的方法入口可以使用subject对象【subject. isPermitted(String)】，也可以使用securityManager对象【securityManager. isPermitted(PrincipalCollection, String)】调用。区别在于subject本身具有身份信息，不需要再传递身份参数，而使用securityManager需要传递身份参数，即PrincipalCollection对象。其实subject最终也是调用securityManager的方法进行鉴权。此外，鉴权必须依赖Realm的配合，因为一些角色、权限信息需要在Realm的doGetAuthorizationInfo方法获取。

Shiro的默认实现，会把登录、授权数据存在subject，而subject存在session。由于“统一鉴权”使用无状态token，因此放弃使用subject，直接使用securityManager进行鉴权。由于角色、权限信息没有进行保存，因此每一次鉴权都需要重新读数据库的role、resource表。