

[**#**](https://pig4cloud.com/#%E7%BB%A7%E6%89%BFbaseresourceserverconfigureradapter%E5%8D%B3%E5%8F%AF%E5%AE%9E%E7%8E%B0%E6%8E%A5%E5%85%A5oauth2) **继承BaseResourceServerConfigurerAdapter即可实现接入oauth2**

[**#**](https://pig4cloud.com/#%E9%87%8D%E5%86%99configure%E6%96%B9%E6%B3%95%E7%9A%84%E6%84%8F%E4%B9%89) **重写configure方法的意义**

BaseResourceServerConfigurerAdapter提供了两种解析用户信息的方法

* 不获取用户详细 只有用户名

protected void notGetUser(ResourceServerSecurityConfigurer resources) {

DefaultAccessTokenConverter accessTokenConverter = new DefaultAccessTokenConverter();

DefaultUserAuthenticationConverter userTokenConverter = new DefaultUserAuthenticationConverter();

accessTokenConverter.setUserTokenConverter(userTokenConverter);

remoteTokenServices.setRestTemplate(lbRestTemplate());

remoteTokenServices.setAccessTokenConverter(accessTokenConverter);

resources.authenticationEntryPoint(resourceAuthExceptionEntryPoint)

.accessDeniedHandler(pigAccessDeniedHandler)

.tokenServices(remoteTokenServices);

}

* 上下文中获取用户全部信息，两次调用userDetailsService，影响性能

private void canGetUser(ResourceServerSecurityConfigurer resources) {

DefaultAccessTokenConverter accessTokenConverter = new DefaultAccessTokenConverter();

DefaultUserAuthenticationConverter userTokenConverter = new DefaultUserAuthenticationConverter();

userTokenConverter.setUserDetailsService(userDetailsService);

accessTokenConverter.setUserTokenConverter(userTokenConverter);

remoteTokenServices.setRestTemplate(lbRestTemplate());

remoteTokenServices.setAccessTokenConverter(accessTokenConverter);

resources.authenticationEntryPoint(resourceAuthExceptionEntryPoint)

.accessDeniedHandler(pigAccessDeniedHandler)

.tokenServices(remoteTokenServices);

}

[**#**](https://pig4cloud.com/#%E6%80%BB%E7%BB%93) **总结**

* 这里也就是为什么SecurityUtils.getUser 有时候返回为空的原因。
* 默认（不重写）是可以换成全部信息，建议不重写；特殊接口为了追求QPS可以重写。