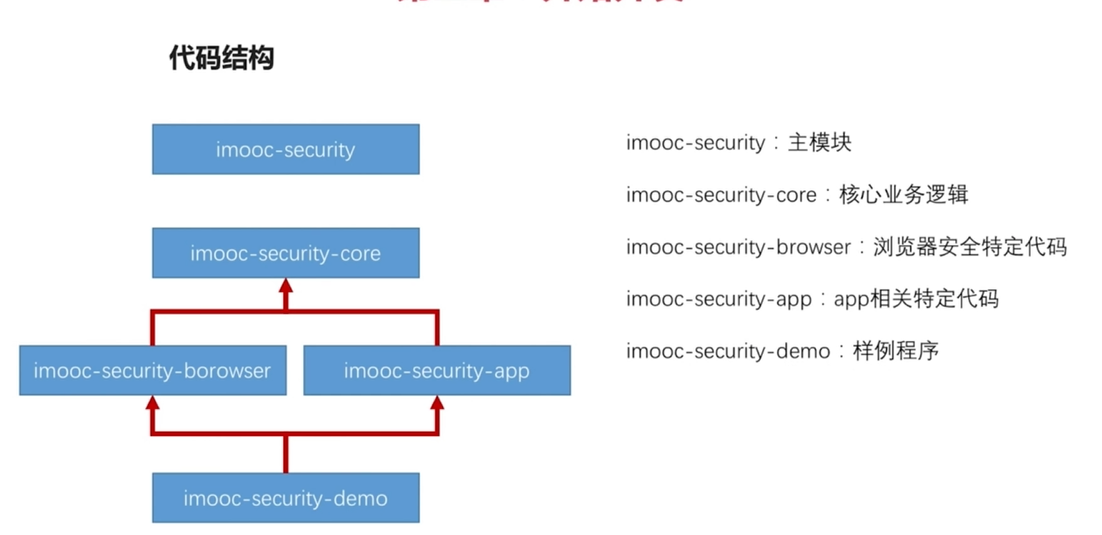
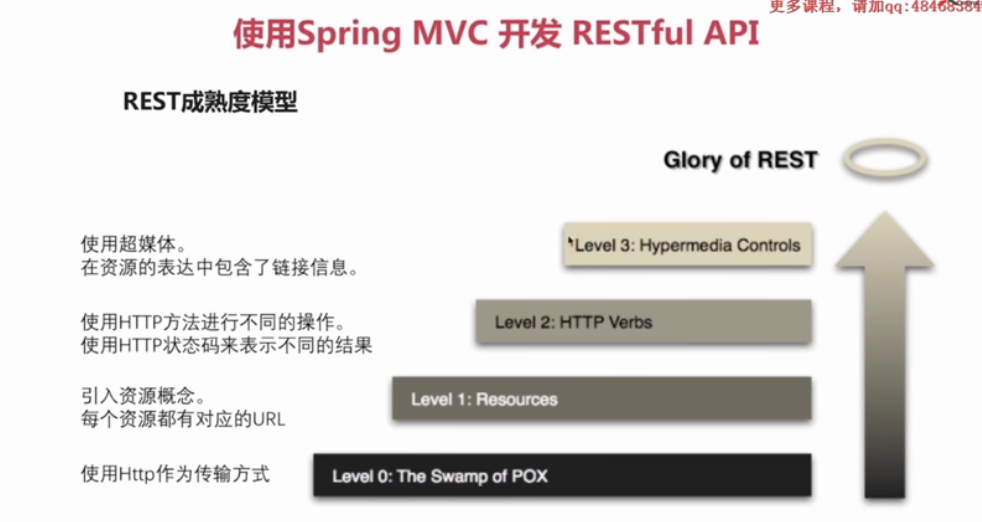
1. 项目结构



1. restful api





1. wiremock 伪造restful 服务

官网下载wiremock 用java –jar命令启动，然后在代码中添加接口和对应的返回值。

1. 常用注解 mockMVC模拟接口 swagger

@RequestParam

@PageableDefault

@PathVariable 映射URL片段到java方法中的参数

1. URL使用正则表达式

JsonView控制json输出

@RequestBody

日期处理 传时间戳

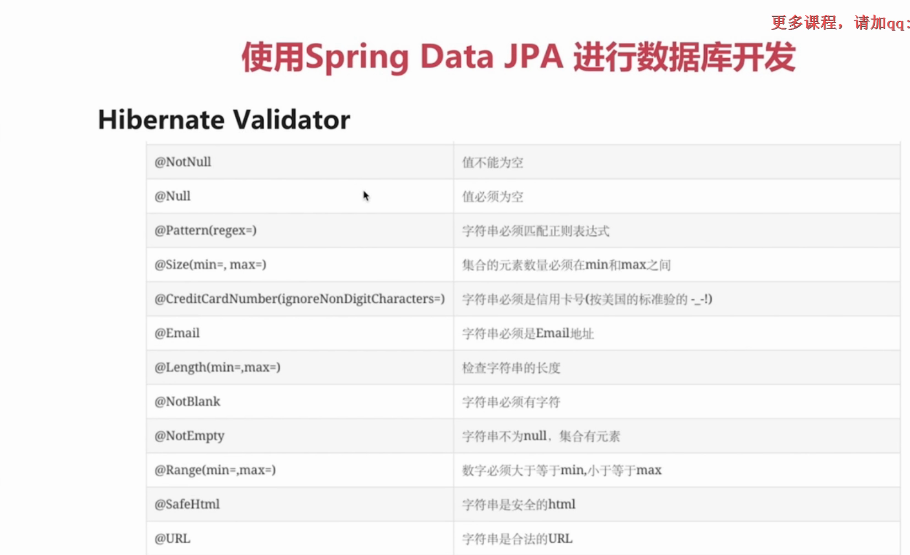
传入时间戳自动转换为日期

返回时自动转为时间戳

Validater

@NotNUll

@Valid



BindingResult

校验后错误的信息对象

GetAllErrors 可以获得所有的错误信息

GetDefaultMessage 获得错误信息提示

1. 自定义注解

处理校验逻辑

1. 自定义enum错误码错误信息

自定义业务异常

自定义异常处理器

1. Filter Interrupt
2. 文件上传和下载

异步 前端直接上传文件，传到后端文件名和id即可

1. spring security



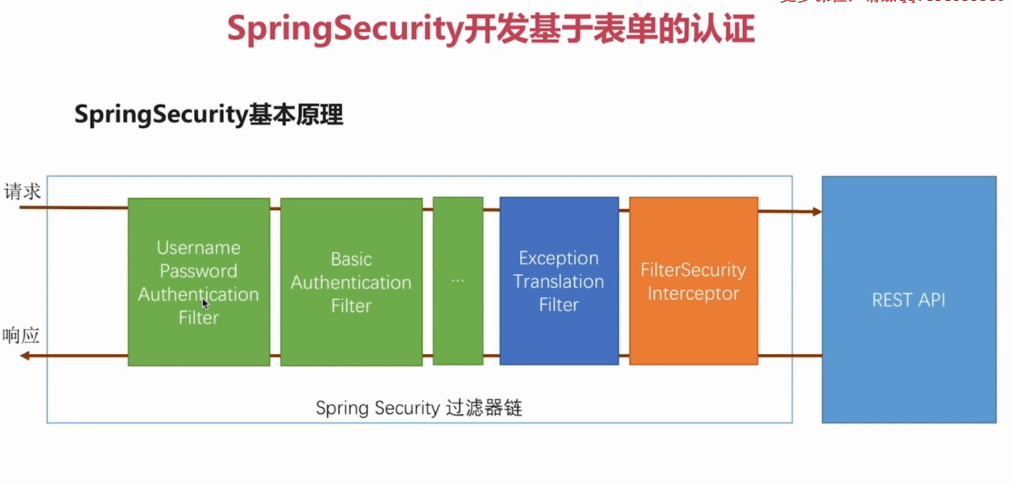
引入spring security

的依赖会自动开始认证机制

用户名是user ， 密码在后台日志中。

覆盖spring security的登录机制。

11.spring security 原理



Spring security 是基于很多过滤器来实现的

绿色的代表登录认证的过滤器

UsernamePassword…Filter 是用户名密码登录的验证过滤器

Egg:http.formLogin()//表单登录

Basic…Filter 是处理默认登录的验证过滤器

http.httpBasic() 默认登录

FilterSecurityInterceptor 是最后一个过滤器

根据配置的条件来判断是否放行，是否允许访问接口。

.and().authorizeRequests().anyRequest().authenticated();//任何请求都需要验证  
根据请求的信息，判断出对应的错误信息。例如密码错误，路径不允许访问等，然后将异常信息抛出

ExceptionTranslationFilter 用来补货抛出的异常，并作出响应，例如跳转登录页。

登录认证的过滤器还有很多钟，例如QQ，微信等第三方登录，都是通过配置认证过滤器来实现的。

这些过滤器都是可以通过配置来实现是否开启的。

@Override  
protected void configure(HttpSecurity http) throws Exception{  
 http.formLogin()//表单登录  
 //http.httpBasic() 默认登录  
 .and().authorizeRequests().anyRequest().authenticated();//任何请求都需要验证  
}

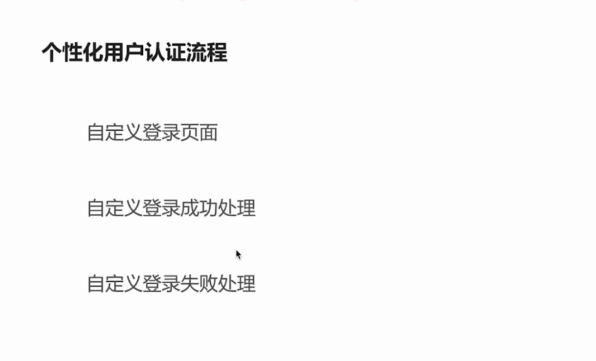
12.自定义用户认证逻辑

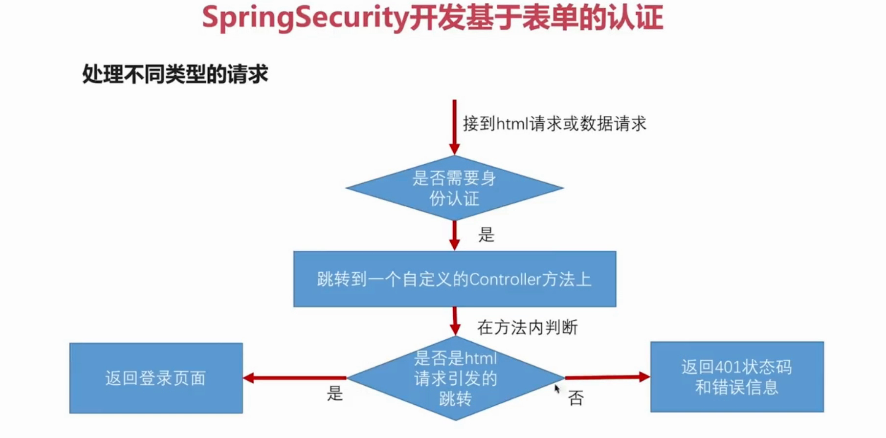


UserDetails loadUserByUsername(String username) throws UsernameNotFoundException;

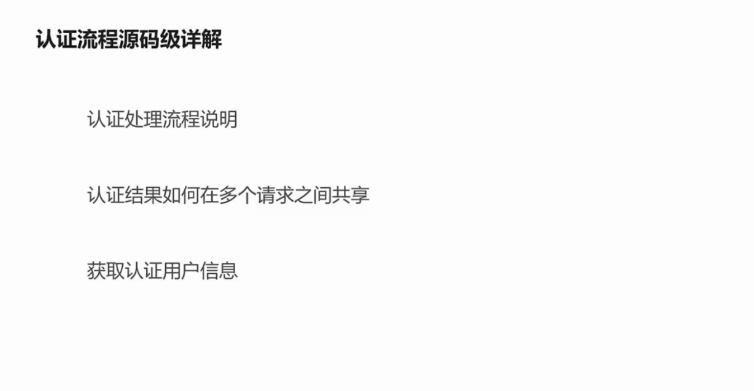
根据username从数据库或其他地方获取用户信息，然后校验逻辑，如果认证成功，将会把用户信息放入到session中，标识认证成功。

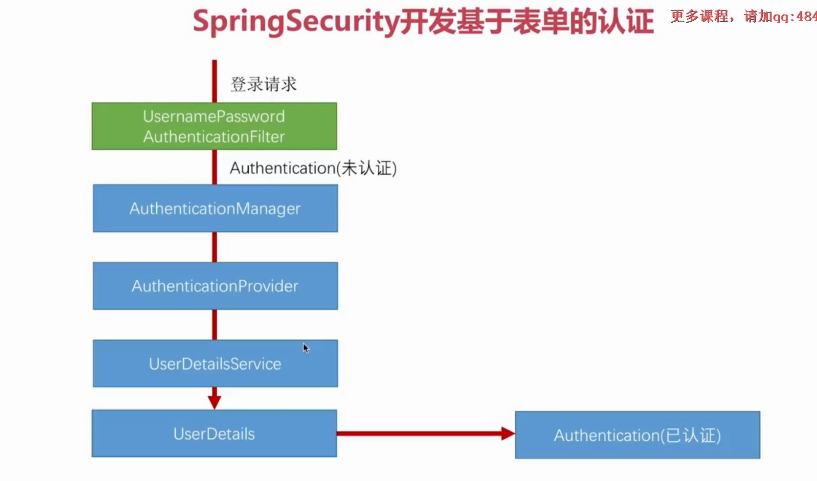


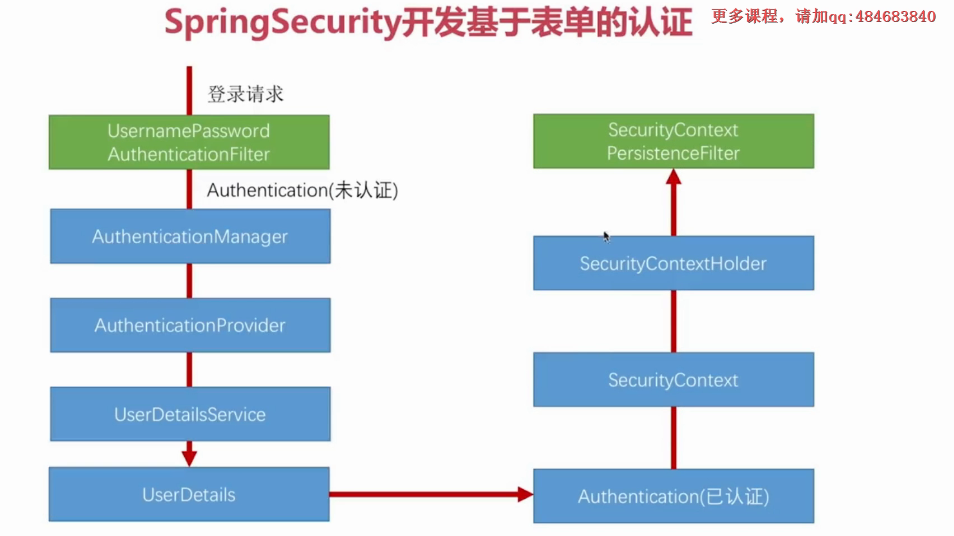












认证在多个请求之间共享

认证信息Authencation放到SecurityContext中

再放到SecurityContextHolder中。

从SecurityContextHolder 中都可以获得当前登录的认证信息。

SecurityContextPersistenceFilter

在过滤器的最顶端，当请求到达时验证session是否有验证信息

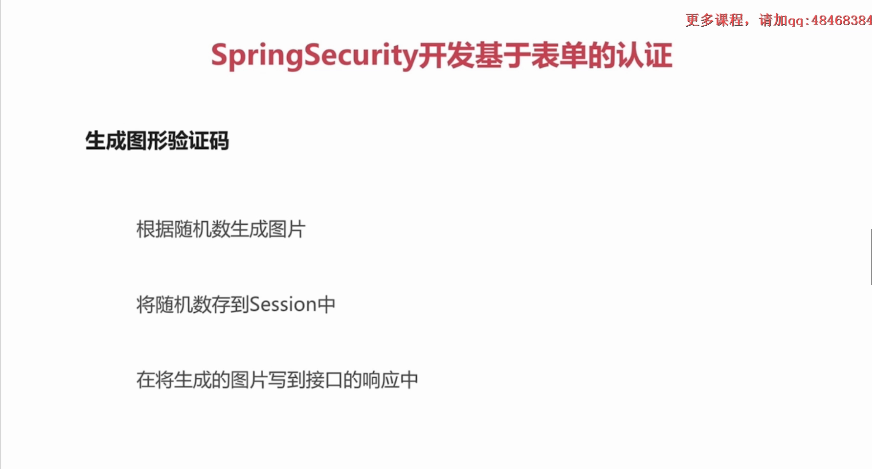
有就放到线程里。在请求结束后拿出线程里的 验证信息放到session中。

不同的请求都可以在同一个session中拿到认证信息。

一个请求之间都可以在SecurityContextHolder中拿到认证信息。

获取用户信息

SecurityContextHolder.*getContext*().getAuthentication();



图片验证码的逻辑

前端通过URL从后端获取对应的包含随机数的图片

后端生成图片，并生成对应的随机数，设置对应的过期时间，将图片放在session中。

然后将生成的图片返回。

登录请求时，在过滤器中做逻辑处理

传入图片的随机码，然后判断，如果没有随机码，提示未输入。如果图片过期，提示图片已过期

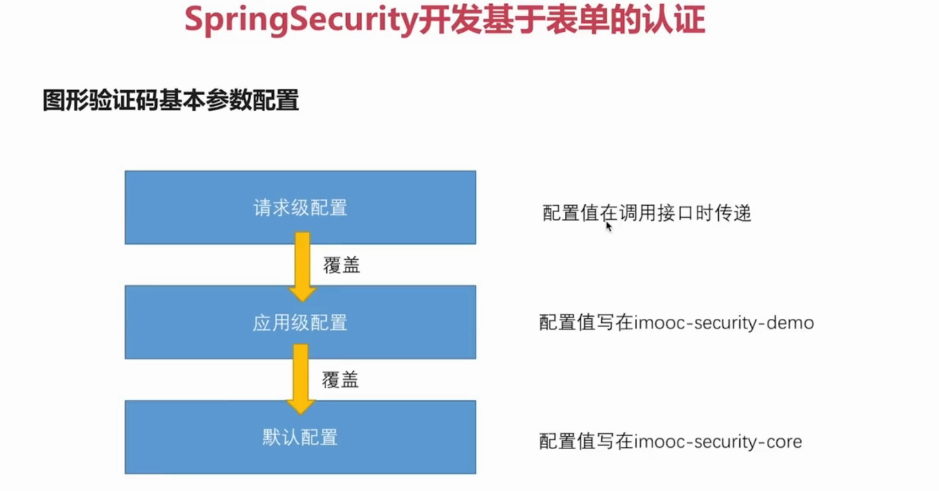
将随机码和session中的图片比较，相同放行。

不同，提示错误。

验证图片验证码

在springsecurity 用户名密码过滤器校验前，添加图片验证码的校验器。

http.addFilterBefore(validateCodeFilter, UsernamePasswordAuthenticationFilter.class).formLogin()



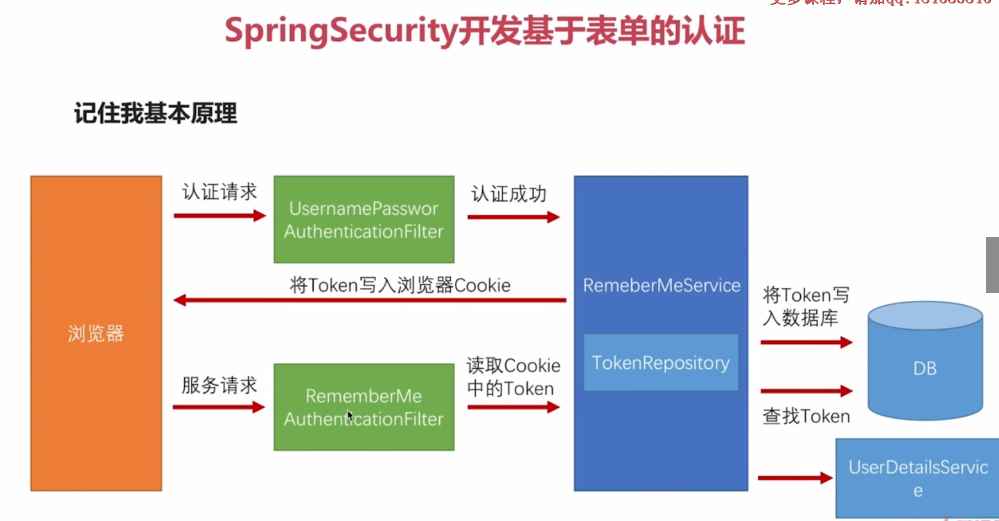
如何将接口的实现做成可配置的。

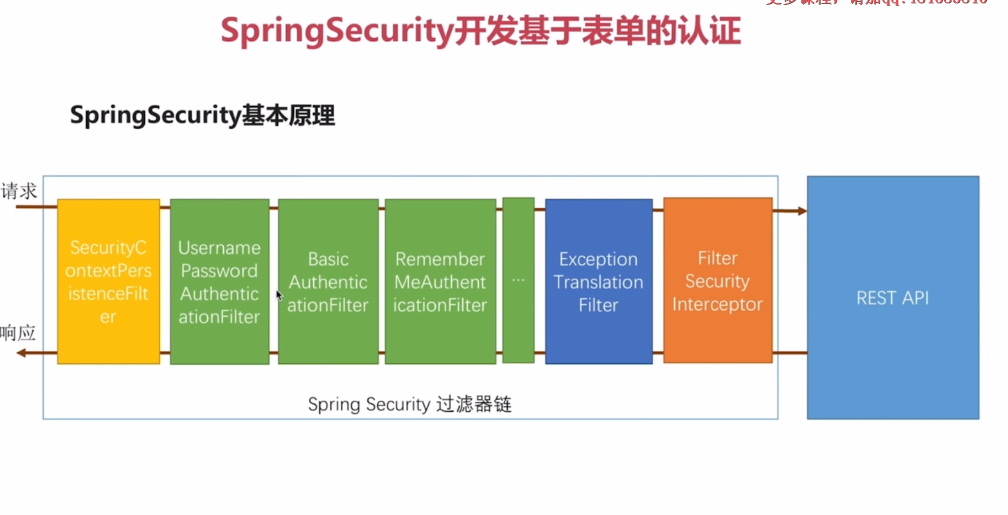
项目中@Bean 默认一个实现类,并加上@ConditionOnMissingBean(name = “123”)

在找不到123的bean时，自己默认创建

也就是说可以自己配置对应的bean

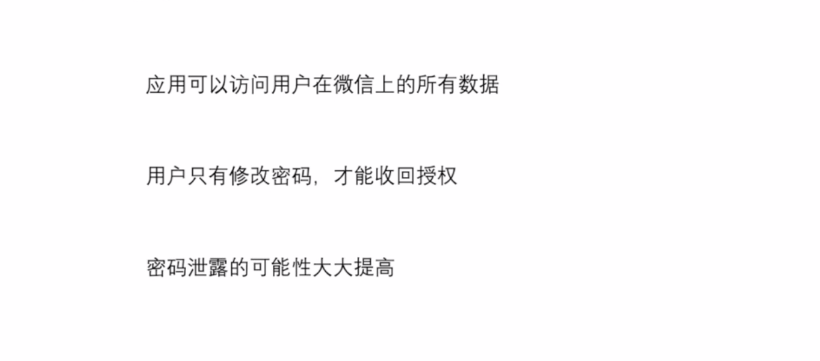
身份识别机制

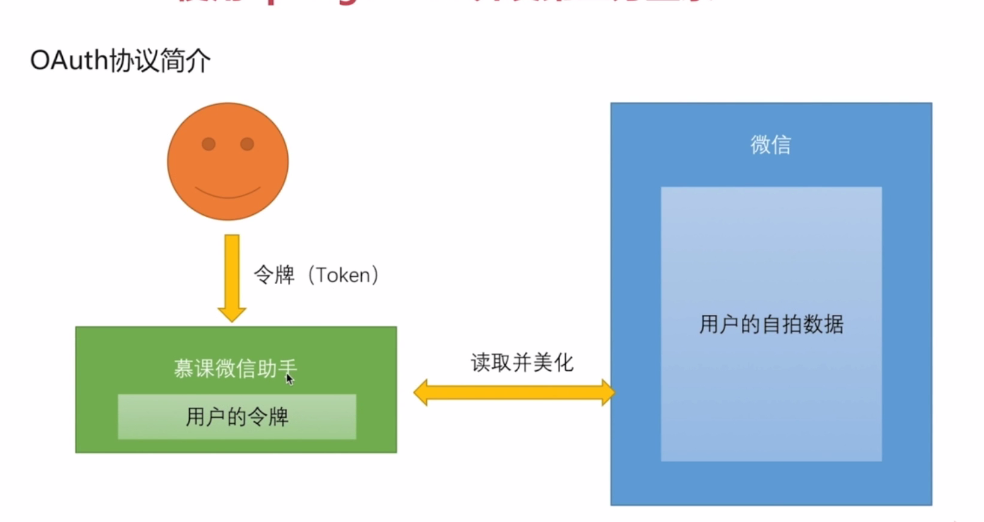




如果直接授权第三方应用账户名和密码

会出现以下问题

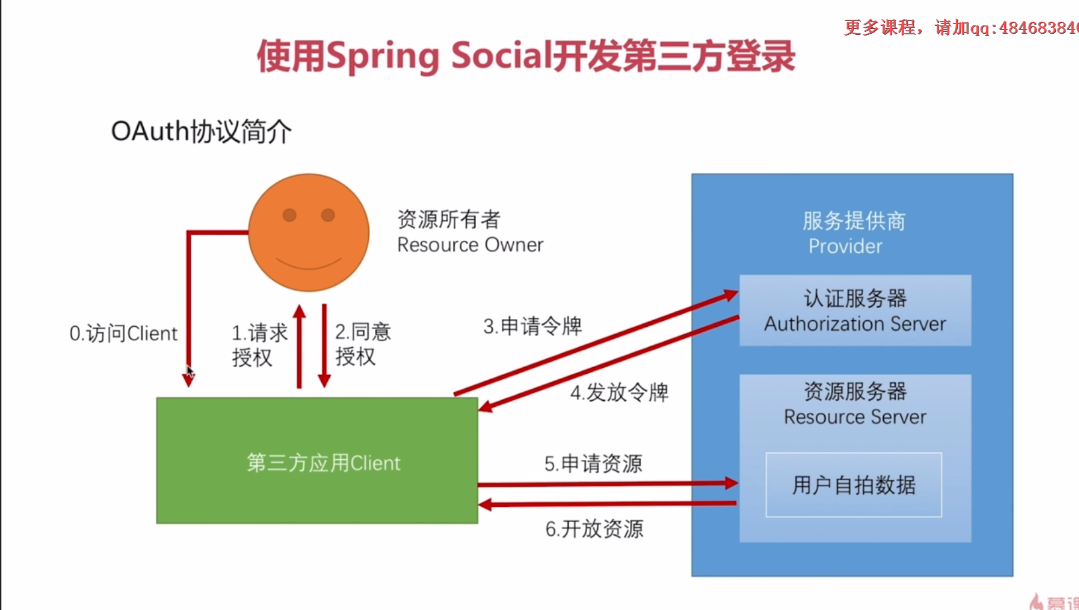




采用用户授权的令牌token来读取数据。

可以设置令牌的权限和有效期。

基本流程





授权码模式

