



下载APP



## 05 | 如何安全、快速地接入OAuth 2.0 ?

2020-07-09 王新栋

OAuth 2.0实战课

[进入课程 >](#)



讲述：李海明

时长 18:40 大小 17.10M



你好，我是王新栋。

在 [第 3 讲](#)，我已经讲了授权服务的流程，如果你还记得的话，当时我特意强调了一点，就是**授权服务将 OAuth 2.0 的复杂性都揽在了自己身上**，这也是授权服务为什么是 OAuth 2.0 体系的核心的原因之一。

虽然授权服务做了大部分工作，但是呢，在 OAuth 2.0 的体系里面，除了资源拥有者是作为用户参与，还有另外两个系统角色，也就是第三方软件和受保护资源服务。那么今天一讲，我们就站在这两个角色的角度，看看它们应该做哪些工作，才能接入到 OAuth 的体系里面呢？



现在，就让我们来看看，作为第三方软件的小兔和京东的受保护资源服务，具体需要着重处理哪些工作吧。

注：另外说明一点，为了脱敏的需要，在下面的讲述中，我只是把京东商家开放平台作为一个角色使用，以便有场景感，来帮助你理解。

## 构建第三方软件应用

我们先来思考一下：如果要基于京东商家开放平台构建一个小兔打单软件的应用，小兔软件的研发人员应该做哪些工作？

是不是要到京东商家开放平台申请注册为开发者，在成为开发者以后再创建一个应用，之后我们就开始开发了，对吧？没错，一定是这样的流程。那么，开发第三方软件应用的过程中，我们需要重点关注哪些内容呢？

我先来和你总结下，这些内容包括 4 部分，分别是：**注册信息、引导授权、使用访问令牌、使用刷新令牌**。

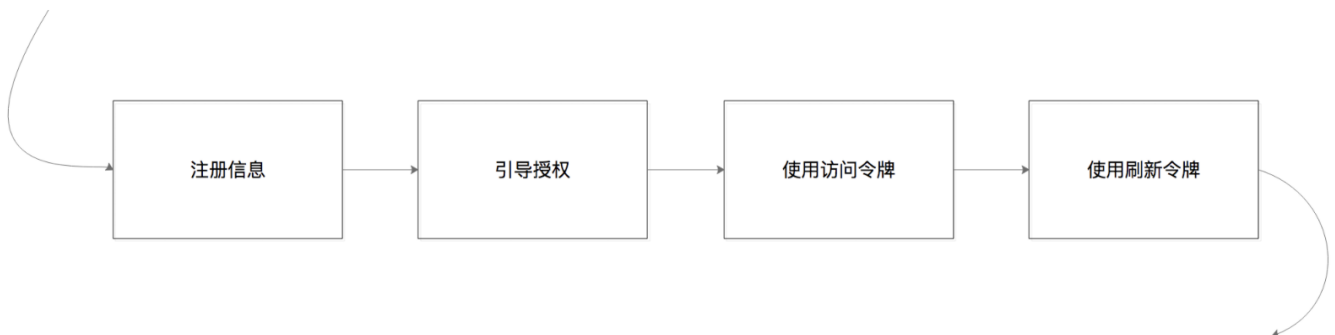


图1 开发第三方软件应用，应该关注的内容

第一点，注册信息。

首先，小兔软件只有先有了身份，才可以参与到 OAuth 2.0 的流程中去。也就是说，小兔软件需要先拥有自己的 app\_id 和 app\_secret 等信息，同时还要填写自己的回调地址 redirect\_uri、申请权限等信息。

这种方式的注册呢，我们有时候也称它为**静态注册**，也就是小兔软件的研发人员提前登录到京东商家开放平台进行手动注册，以便后续使用这些注册的相关信息来请求访问令牌。

## 第二点，引导授权。

当用户需要使用第三方软件，来操作其在受保护资源上的数据，就需要第三方软件来引导授权。比如，小明要使用小兔打单软件来对店铺里面的订单发货打印，那小明首先访问的一定是小兔软件（原则上是直接访问第三方软件，不过我们在后面讲到服务市场这种场景的时候，会有稍微不同），不会是授权服务，更不会是受保护资源服务。

但是呢，小兔软件需要小明的授权，只有授权服务才能允许小明这样做。所以呢，小兔软件需要“配合”小明做的第一件事儿，就是将小明引导至授权服务，如下面代码所示。

那去做什么呢？其实就是让用户为第三方软件授权，得到了授权之后，第三方软件才可以代表用户去访问数据。也就是说，小兔打单软件获得授权之后，才能够代表小明处理其在京东店铺上的订单数据。

[复制代码](#)

```
1 String oauthUrl = "http://localhost:8081/OauthServlet-ch03?reqType=oauth";  
2  
3 response.sendRedirect(toOAuthUrl);
```

## 第三点，使用访问令牌。

**拿到令牌后去使用令牌，才是第三方软件的最终目的。**然后我们看看如何使用令牌。目前OAuth 2.0的令牌只支持一种类型，那就是bearer令牌，也就是我之前讲到的可以是任意字符串格式的令牌。

官方规范给出的使用访问令牌请求的方式，有三种，分别是：

### Form-Encoded Body Parameter（表单参数）

[复制代码](#)

```
1 POST /resource HTTP/1.1  
2 Host: server.example.com  
3 Content-Type: application/x-www-form-urlencoded  
4  
5 access_token=b1a64d5c-5e0c-4a70-9711-7af6568a61fb
```

## URI Query Parameter ( URI 查询参数 )

[复制代码](#)

```
1 GET /resource?access_token=b1a64d5c-5e0c-4a70-9711-7af6568a61fb HTTP/1.1
2 Host: server.example.com
```

## Authorization Request Header Field ( 授权请求头部字段 )

[复制代码](#)

```
1 GET /resource HTTP/1.1
2 Host: server.example.com
3 Authorization: Bearer b1a64d5c-5e0c-4a70-9711-7af6568a61fb
```

也就是说，这三种方式都可以请求到受保护资源服务。那么，我们采用哪种方式最合适呢？

根据 OAuth 2.0 的官方建议，系统在接入 OAuth 2.0 之前信息传递的请求载体是 JSON 格式的，那么如果继续采用表单参数提交的方式，令牌就无法加入进去了，因为格式不符。如果这时采用参数传递的方式呢，整个 URI 会被整体复制，安全性是最差的。而请求头部字段的方式就没有上述的这些“烦恼”，因此官方的建议是采用 Authorization 的方式来传递令牌。

但是，**我建议你采用表单提交，也就是 POST 的方式来提交令牌**，类似如下代码所示。原因是这样的，从官方的建议中也可以看出，它指的是在接入 OAuth 2.0 之前，如果你已经采用了 JSON 数据格式请求体的情况下，不建议使用表单提交。但是，刚开始的时候，只要三方软件 and 平台之间约束好了，大家一致采用表单提交，就没有任何问题了。**因为表单提交的方式在保证安全传输的同时，还不需要去额外处理 Authorization 头部信息。**

[复制代码](#)

```
1 String protectedURL="http://localhost:8082/ProtectedServlet-ch03";
2 Map<String, String> paramsMap = new HashMap<String, String>();
3
4 paramsMap.put("app_id","APPID_RABBIT");
5 paramsMap.put("app_secret","APPSECRET_RABBIT");
6 paramsMap.put("token",accessToken);
7
8 String result = HttpClient.doPost(protectedURL,HttpClient.mapToStr(param
```

第四点，使用刷新令牌。

我在讲授权服务的时候提到过，如果访问令牌过期了，小兔软件总不能立马提示并让小明重新授权一次，否则小明的体验将会非常不好。为了解决这个问题呢，就用到了刷新令牌。

使用刷新令牌的方式跟使用访问令牌是一样的，具体可以参照上面我们讲的访问令牌的方式。关于刷新令牌的使用，你最需要关心的是，什么时候你会来决定使用刷新令牌。

在小兔打单软件收到访问令牌的同时，也会收到访问令牌的过期时间 `expires_in`。一个设计良好的第三方应用，**应该将 `expires_in` 值保存下来并定时检测**；如果发现 `expires_in` 即将过期，则需要利用 `refresh_token` 去重新请求授权服务，以便获取新的、有效的访问令牌。

这种定时检测的方法可以提前发现访问令牌是否即将过期。此外，还有一种方法是“现场”发现。也就是说，比如小兔软件访问小明店铺订单的时候，突然收到一个访问令牌失效的响应，此时小兔软件立即使用 `refresh_token` 来请求一个访问令牌，以便继续代表小明使用他的数据。

综合来看的话，定时检测的方式，需要我们额外开发一个定时任务；而“现场”发现，就没有这种额外的工作量啦。具体采用哪一种方式，你可以结合自己的实际情况。不过呢，我还是建议你采用定时检测这种方式，因为它可以带来“提前量”，以便让我们有更好的主动性，而现场发现就有点被动了。

说到这里，我要再次提醒你注意的是，**刷新令牌是一次性的，使用之后就会失效**，但是它的有效期会比访问令牌要长。这个时候我们可能会想到，如果刷新令牌也过期了怎么办？在这种情况下，我们就需要将刷新令牌和访问令牌都放弃，相当于回到了系统的初始状态，只能让用户小明重新授权了。

到这里，我们来总结下，在构建第三方应用时，你需要重点关注的就是注册、授权、访问令牌、刷新令牌。只要你掌握了这四部分内容，在类似京东这样的开放平台上开发小兔软件，就不再是什么困难的事情了。



## 服务市场中的第三方应用软件

在构建第三方应用的引导授权时，我们说用户第一次“触摸”到的一定是第三方软件，但这并不是绝对的。这个不绝对，就发生在服务市场这样的场景里。

那什么是服务市场呢？说白了，就是你开发的软件，比如小兔打单软件、店铺装修软件等，都发布到这样一个“市场”里面售卖。这样，当用户购买了这些软件之后，就可以在服务市场里面看到有个“立即使用”的按钮。点击这个按钮，用户就可以直接访问自己购买的第三方软件了。

比如，京东的京麦服务市场里有个“我的服务”目录，里面就存放了我购买的打单软件。小明就可以直接点击“立即使用”，继而进入小兔打单软件，如下图所示。

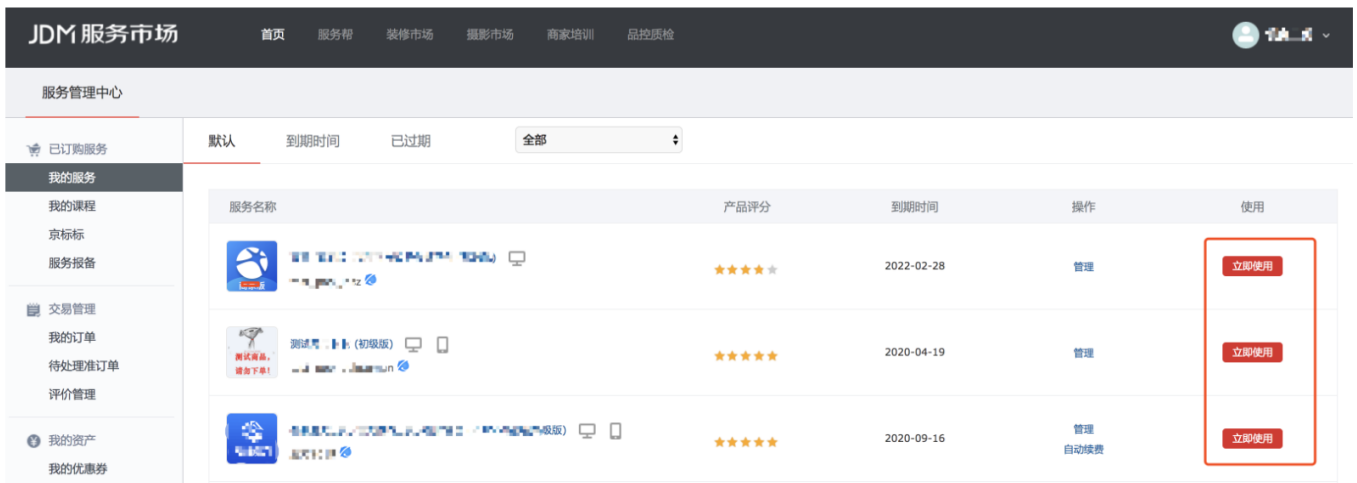


图2 京麦服务市场“我的服务”

那么，这里需要注意的是，作为第三方开发者来构建第三方软件的时候，在授权码环节除了要接收授权码 code 值之外，还要接收用户的订购相关信息，比如服务的版本号、服务代码标识等信息。

好了，以上就是关于构建第三方软件时需要注意的一些细节问题了。接下来，我们再谈谈构建受保护资源服务的时候，又需要重点处理哪些工作呢。

## 构建受保护资源服务

你先想一想，实际上在整个开放授权的环境中，受保护资源最终指的还是 Web API，比如说，访问头像的 API、访问昵称的 API。对应到我们的打单软件中，受保护资源就是订单查

询 API、批量查询 API 等。

在互联网上的系统之间的通信，都是以 Web API 为载体的形式进行。因此呢，当我们说受保护资源被授权服务保护着时，实际上说的是授权服务最终保护的是这些 Web API。我们在构建受保护资源服务的时候，除了基本的要检查令牌的合法性，还需要做些什么呢？我认为**最重要的就是权限范围了**。

在我们处理受保护资源服务中的逻辑的时候，校验权限的处理会占据很大的比重。你想啊，访问令牌递过来，你肯定要多看看令牌到底能操作哪些功能、又能访问哪些数据吧。现在，我们把这些权限的类别总结归纳下来，最常见的大概有以下几类。

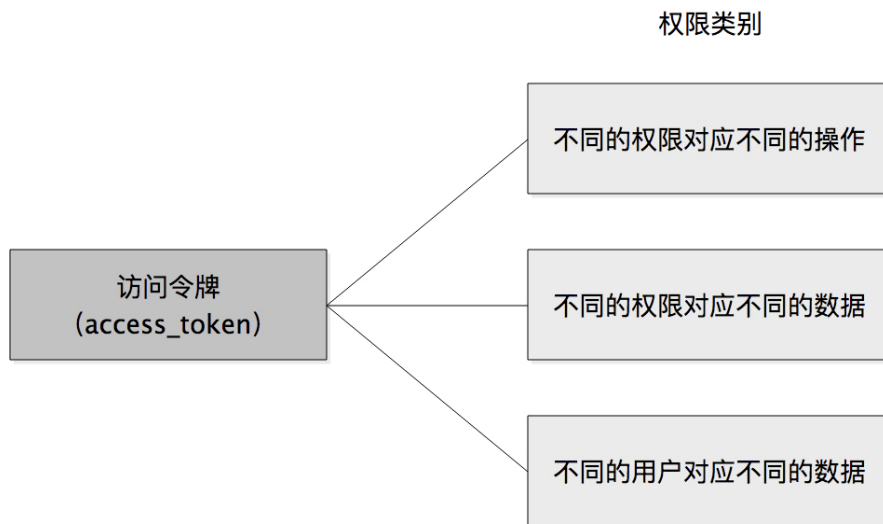


图3 3类权限类别

接下来，我和你具体说说这些权限是如何使用的。

### 1. 不同的权限对应不同的操作。

这里的操作，其实对应的是 Web API，比如目前京东商家开放平台提供有查询商品 API、新增商品 API、删除商品 API 这三种。如果小兔软件请求过来的一个访问令牌 `access_token` 的 `scope` 权限范围只对应了查询商品 API、新增商品 API，那么包含这个 `access_token` 值的请求，就不能执行删除商品 API 的操作。

复制代码

```
1 //不同的权限对应不同的操作
2 String[] scope = OAuthServlet.tokenScopeMap.get(accessToken);
3
```

```
4 StringBuffer sbuf = new StringBuffer();
5 for(int i=0;i<scope.length;i++){
6     sbuf.append(scope[i]).append("|");
7 }
8
9 if(sbuf.toString().indexOf("query">0){
10     queryGoods("");
11 }
12
13 if(sbuf.toString().indexOf("add">0){
14     addGoods("");
15 }
16
17 if(sbuf.toString().indexOf("del">0){
18     delGoods("");
19 }
```

## 2. 不同的权限对应不同的数据。

这里的数据，就是指某一个 API 里包含的属性字段信息。比如，有一个查询小明信息的 API，返回的信息包括 Contact ( email、phone、qq )、Like ( Basketball、Swimming )、Personal Data ( sex、age、nickname )。如果小兔软件请求过来的一个访问令牌 access\_token 的 scope 权限范围只对应了 Personal Data，那么包含该 access\_token 值的请求就不能获取到 Contact 和 Like 的信息，关于这部分的代码，实际跟不同权限对应不同操作的代码类似。

看到这里，你就明白了，这种权限范围的粒度要比“不同的权限对应不同的操作”的粒度要小。这正是遵循了最小权限范围原则。

## 3. 不同的用户对应不同的数据。

这种权限是什么意思呢？其实，这种权限实际上只是换了一种维度，将其定位到了用户上面。

一些基础类信息，比如获取地理位置、获取天气预报等，不会带有用户归属属性，也就是说这些信息并不归属于某个用户，是一类公有信息。对于这样的信息，平台提供出去的 API 接口都是“中性”的，没有用户属性。



但是，更多的场景却是基于用户属性的。还是以小兔打单软件为例，商家每次打印物流面单的时候，小兔打单软件都要知道是哪个商家的订单。这种情况下，商家为小兔软件授权，小兔软件获取的 `access_token` 实际上就包含了商家这个用户属性。

京东商家开放平台的受保护资源服务每次接收到小兔软件的请求时，都会根据该请求中 `access_token` 的值找到对应的商家 ID，继而根据商家 ID 查询到商家的订单信息，也就是不同的商家对应不同的订单数据。

[复制代码](#)

```
1 //不同的用户对应不同的数据
2 String user = OAuthServlet.tokenMap.get(accessToken);
3 queryOrders(user);
```

在上面讲三种权限的时候，我举的例子实际上都属于一个系统提供了查询、添加、删除这样的所有服务。此时你可能会想到，现在的系统不已经是分布式系统环境了么，如果有很多个受保护资源服务，比如提供用户信息查询的用户资源服务、提供商品查询的商品资源服务、提供订单查询的订单资源服务，那么每个受保护资源服务岂不是都要把上述的权限范围校验执行一遍吗，这样不就会有大量的重复工作产生么？

在这里，我特别高兴你能想到这一点。为了应对这种情况，我们应该有一个统一的网关层来处理这样的校验，所有的请求都会经过 API GATEWAY 跳转到不同的受保护资源服务。这样呢，我们就不需要在每一个受保护资源服务上都做一遍权限校验的工作了，而只需要在 API GATEWAY 这一层做权限校验就可以了。系统结构如下图所示。

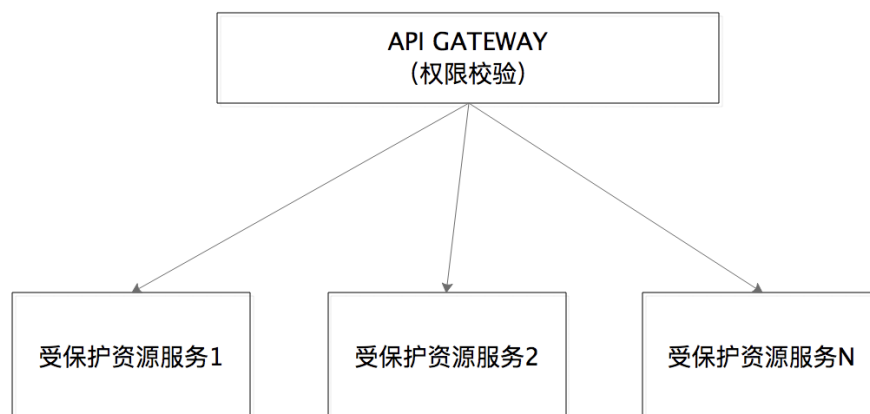


图4 由统一的网关层处理权限校验

## 总结

截止到这一讲呢，我们已经把 OAuth 2.0 中授权码相关的流程所涉及到的内容都讲完了。通过 02 到 05 这 4 讲，你可以很清晰地理解授权码流程的核心原理了，也可以弄清楚如何使用以及如何接入这一授权流程了。

我在本讲开始的时候，提到 OAuth 2.0 的复杂性实际上都给了授权服务来承担，接着我从第三方软件和受保护资源的角度，分别介绍了这两部分系统在接入 OAuth 2.0 的时候应该注意哪些方面。总结下来，我其实希望你能够记住以下两点。

1. 对于第三方软件，比如小兔打单软件来讲，**它的主要目的就是获取访问令牌，使用访问令牌**，这当然也是整个 OAuth 2.0 的目的，就是让第三方软件来做这两件事。在这个过程中需要强调的是，第三方软件在使用访问令牌的时候有三种方式，我们建议在平台和第三方软件约定好的前提下，**优先采用 Post 表单提交的方式**。
2. 受保护资源系统，比如小兔软件要访问开放平台的订单数据服务，它需要注意的是权限的问题，这个权限范围主要包括，**不同的权限会有不同的操作，不同的权限也会对应不同的数据，不同的用户也会对应不同的数据**。

## 思考题

如果使用刷新令牌 refresh\_token 请求回来一个新的访问令牌 access\_token，按照一般规则授权服务上旧的访问令牌应该要立即失效，但是如果在这之前已经有使用旧的访问令牌发出去请求，不就受到影响了吗，这种情况下应该如何处理呢？

欢迎你在留言区分享你的观点，也欢迎你把今天的内容分享给其他朋友，我们一起交流。

提建议

更多课程推荐

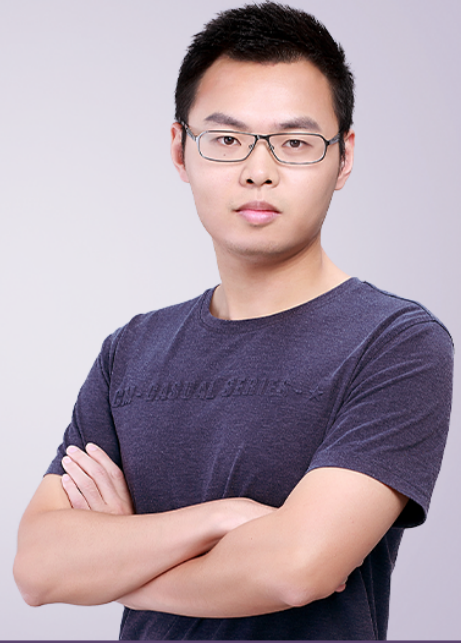
# 设计模式之美

前 Google 工程师手把手教你写高质量代码

王争

前 Google 工程师

《数据结构与算法之美》专栏作者



涨价倒计时 🕒

限时秒杀 **¥149**，7月31日涨价至 **¥299**

© 版权归极客邦科技所有，未经许可不得传播售卖。页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇 04 | 在OAuth 2.0中，如何使用JWT结构化令牌？

下一篇 06 | 除了授权码许可类型，OAuth 2.0还支持什么授权流程？

## 精选留言 (17)

写留言



马以

2020-07-09

思考题：方法1：根据服务的响应时间，两个access\_token要有一个重叠窗口期，过了重叠窗口期，旧的token就失效了；方法2：或者说和mysql数据库事务控制一样，在用refresh\_token 换取 access\_token 这里记录一个节点（水位线）；如果请求在这个节点之前，那么旧的token就还有效，如果请求在这个节点之后，旧的token就失效；

展开 ∨



4



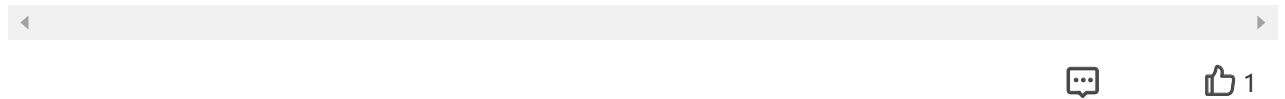
iLeGeND

2020-07-09

老师以后要不要出一门spring security的课

展开 ∨

作者回复: 暂时 还没有计划哈

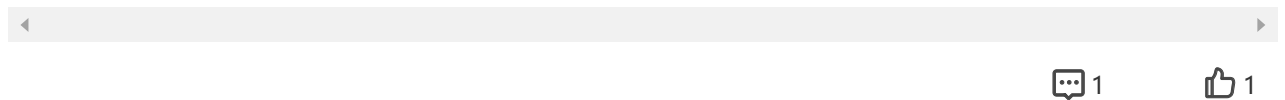


hhhh

2020-07-09

“建议在平台和第三方软件约定好的前提下，优先采用 Post 表单提交的方式。”，如果这样，每个请求都需要带有token，那三方软件和开放平台只能使用post方式通信了吗？

作者回复: 是的



自在如风

2020-07-22

查询通过redis去查。在调用refresh接口之后,新的access\_token插入到redis中,同时设置过期时间为access\_token的过期时间。同时老的access\_token可以重新插入redis中,设置过期时间为1min,这样应该能解决那个思考题的问题

展开 ∨

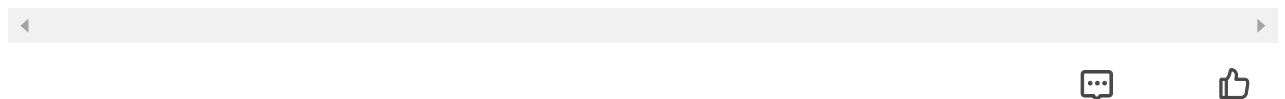


在路上

2020-07-19

老师 令牌采用定时刷新的方式，令牌永远不会过期，除非用户主动操作，这样理解是都正确。

作者回复: 刷新令牌也有有效期，所以不会有访问令牌永远不过期的情况发生的。



土豆

2020-07-16

老师我想请下，redistoken的缺点是只能单点，不支持多点，jwt是在认证服务器加密一次，在各个资源服务器随时解密，没有失效时间，如果配置多资源服务器，是不是要考虑私钥的区分，两个都有缺点，现在主流的使用方式是哪种，希望老师解惑

展开 ∨

**piboye**

2020-07-15

缩短之前access token和refresh token的超时时间为几秒，这样可以让换token失败的三方软件有机会重试。重试的时候返回上次成功的新token

展开 ∨

**suhuijie**

2020-07-15

对于服务市场的第三方软件，流程如何处理的，那块讲述的不清晰

作者回复: 服务市场这个是补充了一块内容，服务市场的定义咱们课程里面就是说小兔打单软件放在上面售卖，小明购买了之后就可以直接通过服务市场的【我的服务】使用，因为咱们之前一直是讲小明直接访问小兔的地址。

“作为第三方开发者来构建第三方软件的时候，在授权码环节除了要接收授权码 code 值之外，还要接收用户的订购相关信息，比如服务的版本号、服务代码标识等信息。”

OAuth 2.0 的授权码流程不会变，是跟我们已知的一样，只不过这次是从【我的服务】点进去的，补充上面版本号、服务代码等信息是服务相关的属性。

**hom**

2020-07-14

请问下老师：刷新令牌的作用是什么？为什么不直接延长访问令牌的初始化时效，而是需要去手工刷新呢？

作者回复: 出于安全的考虑我们让访问令牌的有效时间一般比较短，刷新令牌的作用是在访问令牌失效的情况下请求获取新的访问令牌。

**一步**

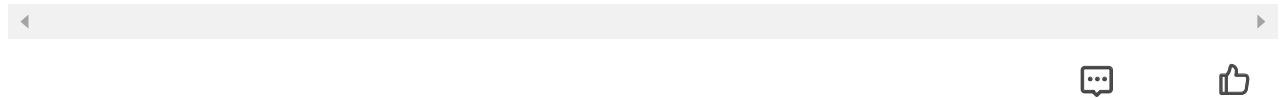
2020-07-11

刷新令牌是一次性的，使用之后就会失效

在获取访问令牌的时候同时也会把 刷新令牌 返回，后面可以使用 刷新令牌 重新获取访问令牌，然后也会同时获取新的刷新令牌，这样的话不是可以一直获取访问令牌了吗？

展开 ∨

作者回复: 刷新令牌的有效期不会变, 而且它本来就比访问令牌的有效期长; 刷新令牌的值变是为了安全, 因为你已经使用过一次了。

**杨浩宇**

2020-07-11

对于思考题, 我觉得比较简单的方式是刷新完后的令牌和刷新前的令牌保持一致, 只是延长了服务器这边的生命周期, 但是对于jwt令牌, 就只能是缩短access\_token的生命周期了

**杨浩宇**

2020-07-11

对于现在前后端分离的应用, 用post方式来使用访问令牌, 是不是就没有办法保护前端页面了? 而且是不是还得要求后台的接口设计上更新删除操作也得用post方式才可以?

**罗乾林**

2020-07-09

课后题:

access\_token失效后请求必定失败, 比较本地access\_token是否更新或是否正在刷新access\_token, 若已更新, 采用新的access\_token 重新请求。若正在刷新将请求放入队列作为挂起任务, 待access\_token更新后重新发出请求

展开 ∨

**zifeng**

2020-07-09

之前用刷新令牌, 策略是access\_token快要过期使用refresh\_token返回新的access\_token, 考虑系统调用的延迟, 一般快要过期的时间窗口设为30秒。

展开 ∨

**暖色浮余生**

2020-07-09

OAuth2 的 scope权限检验和springsecurity的权限检验有什么区别呢, 这两个不是通常



放在一块么，做的东西应该差不多吧

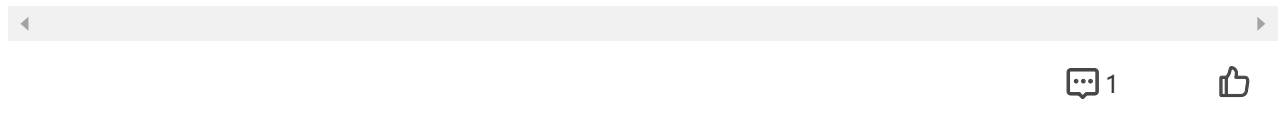
展开 ∨

作者回复: OAuth2，是一种【协议】，安全授权协议；

Spring Security OAuth2，是一种技术【框架】，它是依据OAuth2协议开发出来的；

Spring Security，是一种技术【框架】，一种提供声明式的安全访问控制解决方案的安全框架，跟OAuth没有直接关系；

scope权限检验，大家思想方式都是解决权限访问问题。



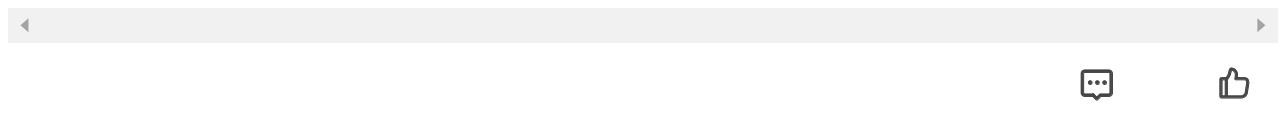
**bigben**

2020-07-09

第三方调用授权服务也是通过api网关的吧？

展开 ∨

作者回复: 开放平台对外主要有两个系统一个是API网关，一个是授权系统。授权（生成TOKEN）是通过请求授权系统，鉴权（识别TOKEN）是调用到API网关，因为使用TOKEN的时候就是发生在通过API网关调用开放API的时候，API网关再去跟授权系统通信。



**bigben**

2020-07-09

对于思考题，旧的access\_token,还是有效的，可以缩短其有效期，但如果是jwt实现的话，只能等待它过期了

展开 ∨

