**一、应用场景举例**

1、用户授权，网盘允许第三方打印服务“云打印”可以访问用户的照片

**二、专用名词**

（1） Third-party application：第三方应用，客户端；比如例子中云打印

（2）HTTP service：http服务提供商，或者就叫服务提供商，比如例子中的网盘

（3）Resource Owner：用户

（4）User Agent：用户代理，用户通过浏览器使用云打印，那么用户代理就是浏览器

（5）Authorization server：认证服务器，例子中的网盘提供的专门用来处理认证的服务器。

（6）Resource server：资源服务器，可能跟上面的是同一台服务器，也可能是分布在不同的服务器上

**三、思路**

1、服务提供商和第三方应用间加一个授权层，第三方应用不会获得用户的密码，而是获取授权层给出的token,有权限范围和使用期限

**四、运行流程**

1）用户访问第三方应用（客户端），第三方应用希望用户授权

2）用户同意给予第三方应用授权

3）第三方应用根据用户授权，向认证服务器请求token

4）认证服务器认证确认无误后，给第三方应用token

5）第三方的应用用这个token去资源服务器那里申请资源

6）资源服务器验证token，确认无误后给出资源

**五、客户端获取授权**

用户同意给予客户端授权，客户端获取到用户授权，即上面的第二步，是重点

主要有一下几种方式：1）授权码模式 ； 2）简化模式；3）密码模式；4）客户端模式

其中第一种授权码模式是功能最完整，流程最严密；被应用最多

**六、授权码模式**

特点：通过客户端的后台服务，与‘服务提供商’的认证服务器进行互动

描述下客户端获取token的过程

1）用户访问客户端，客户端将用户导向认证服务器，并加一个重定向uri：redirect\_uri

客户端申请认证的URI，包含以下参数：

* response\_type：表示授权类型，必选项，此处的值固定为"code"，表示授权码方式
* client\_id：表示客户端的ID，必选项
* redirect\_uri：表示重定向URI，可选项
* scope：表示申请的权限范围，可选项
* state：表示客户端的当前状态，可以指定任意值，认证服务器会原封不动地返回这个值。

请求举例：GET /authorize?response\_type=code&client\_id=s6BhdRkqt3&state=xyz &redirect\_uri=https%3A%2F%2Fclient%2Eexample%2Ecom%2Fcb HTTP/1.1 Host: server.example.com

2）用户同意认证服务器给客户端的授权；认证服务器就生成一个授权码，浏览器将用户导向上一步客户端提供的redirect\_uri

这一步中认证服务器给客户端的回复需包括：授权码、state(如果客户端发过来的话，需要原样返回)

其中授权码时效很短，一般10分钟，并且只能使用一次，在第三步中使用时与client\_id redirect\_uri需要保持匹配

回复举例：HTTP/1.1 302 Found Location: <https://client.example.com/cb>?code=SplxlOBeZQQYbYS6WxSbIA &state=xyz

3）客户端用这个授权码，向认证服务器申请token，并且也还附上redirect\_uri

客户端去向认证服务器申请token的请求中需要包括：

* grant\_type：表示使用的授权模式，必选项，此处的值固定为"authorization\_code"。
* code：表示上一步获得的授权码，必选项。
* redirect\_uri：表示重定向URI，必选项，且必须与步骤1中的该参数值保持一致。
* client\_id：表示客户端ID，必选项。

请求举例：POST /token HTTP/1.1 Host: server.example.com Authorization: Basic czZCaGRSa3F0MzpnWDFmQmF0M2JW Content-Type: application/x-www-form-urlencoded grant\_type=authorization\_code&code=SplxlOBeZQQYbYS6WxSbIA &redirect\_uri=https%3A%2F%2Fclient%2Eexample%2Ecom%2Fcb

4）认证服务器验证了授权码和redirect\_uri后，给客户端发出access\_token和refresh\_token

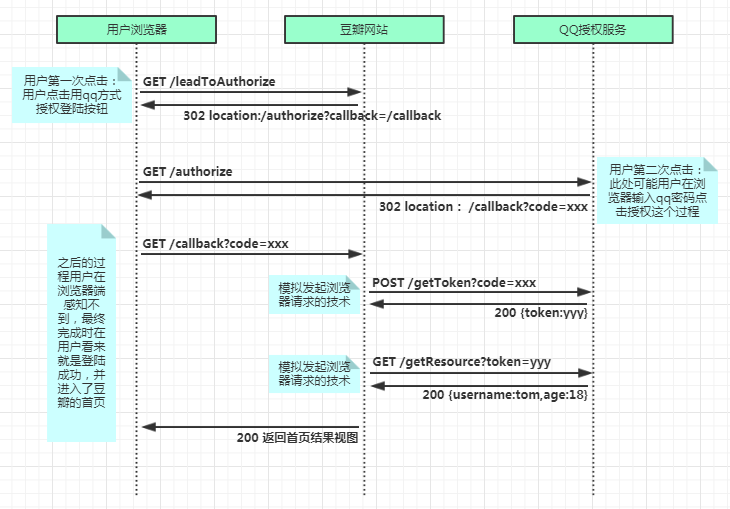
认证服务器发出token的回复里需要有如下参数：

* access\_token：表示访问令牌，必选项。
* token\_type：表示令牌类型，该值大小写不敏感，必选项，可以是bearer类型或mac类型。
* expires\_in：表示过期时间，单位为秒。如果省略该参数，必须其他方式设置过期时间。
* refresh\_token：表示更新令牌，用来获取下一次的访问令牌，可选项。
* scope：表示权限范围，如果与客户端申请的范围一致，此项可省略。

回复举例：可以看出相关参数使用JSON格式发送（Content-Type: application/json）。此外，HTTP头信息中明确指定不得缓存。

HTTP/1.1 200 OK Content-Type: application/json;charset=UTF-8 Cache-Control: no-store Pragma: no-cache { "access\_token":"2YotnFZFEjr1zCsicMWpAA", "token\_type":"example", "expires\_in":3600, "refresh\_token":"tGzv3JOkF0XG5Qx2TlKWIA", "example\_parameter":"example\_value" }

最后以豆瓣登录又qq授权的图来演示一下；第一次点击就是用户点击豆瓣登录页面上的第三方登录的qq的小图标；第二次点击是qq登录页面上的“授权并登录”按钮；更多可查看文档：



七、简化模式

与授权码模式不同，简化模式不通过第三方应用（客户端）后台去和认证服务器打交道，直接由用户代理（例子中的浏览器）向认证服务器申请令牌。所有步骤在浏览器中完成，令牌对访问者是可见的，且客户端不需要认证。