

# 理解 OAuth2 协议

*Tencent*  
*Enterprise*

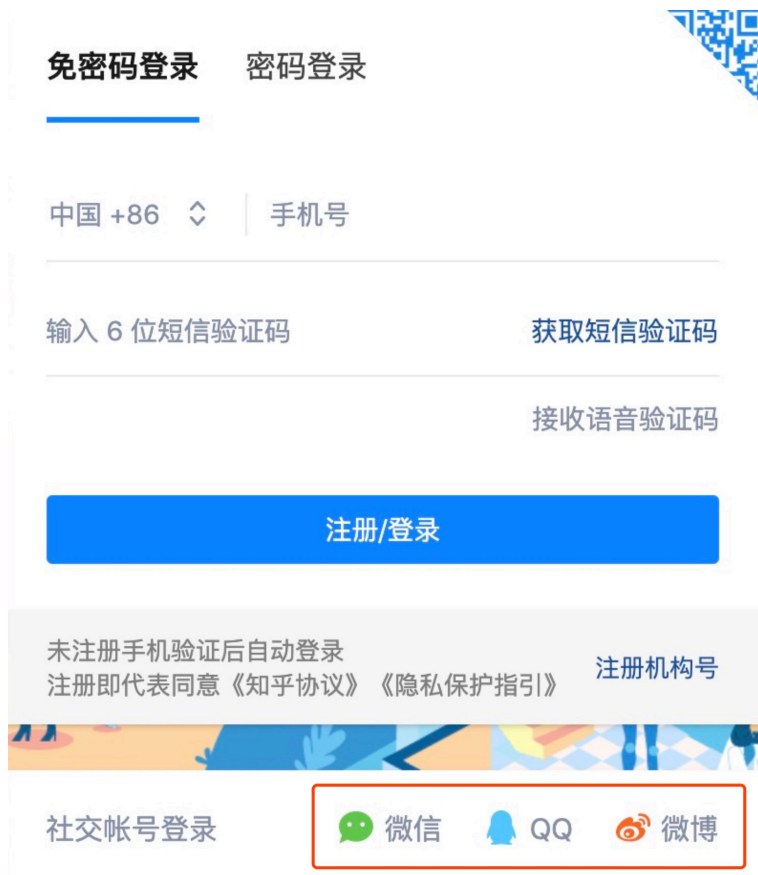


## 目录

- 理解 OAuth2 协议（ open authorization ）
- 单点登录与第三方登录主流的登录方案

# 什么是第三方登录

很多网站登录时，允许使用第三方网站的身份，这称为"第三方登录"。



免密码登录 密码登录

中国 +86 手机号

输入 6 位短信验证码 获取短信验证码

接收语音验证码

注册/登录

未注册手机验证后自动登录 注册机构号

注册即代表同意《知乎协议》《隐私保护指引》

社交帐号登录

微信 QQ 微博



登录 注册

请输入登录手机号/邮箱

请输入密码

☒ 7天内自动登录 找回密码 无法登录

登录

手机短信登录

微信 QQ 微博



# 理解 OAuth2 协议

ruby-china.org/account/sign\_in

relopment reading learning welfare movie

Ruby China

社区

招聘

Wiki

酷站

搜索本站内容

注册

登录

用户名 / Email

密码

☐ 记住登录状态 (60 天)

登录

用其他平台的帐号登录

GitHub

注册

忘记了密码?

未收到解锁邮件?



关于 RubyConf Ruby 镜像 RubyGems 镜像 Status 活跃会员 组织 API 贡献者  
由众多爱好者共同维护的 Ruby 中文社区, 本站使用 Homeland 构建, 并采用 Docker 部署。

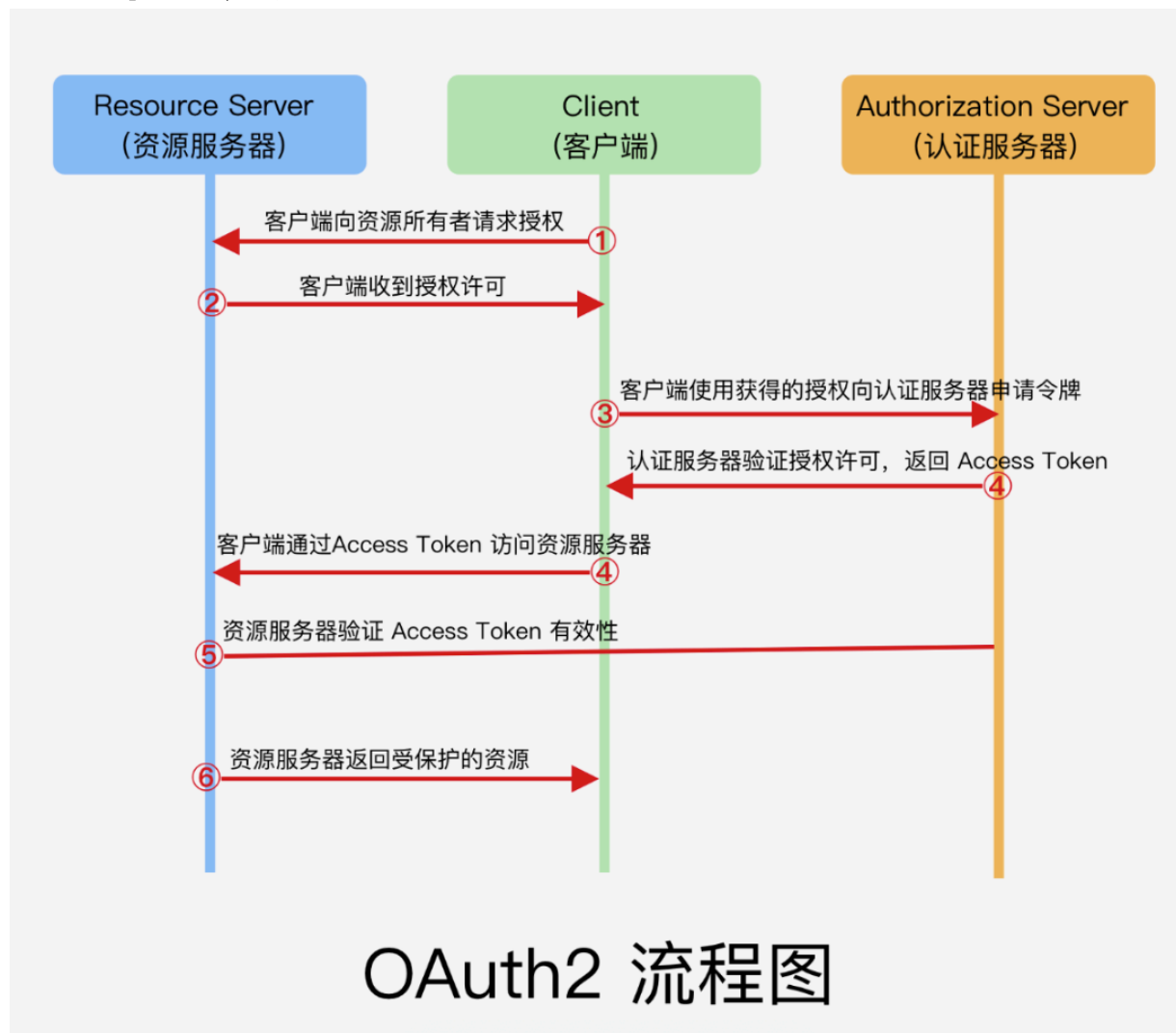
服务器由 Ucloud 赞助 CDN 由 又拍云 赞助



iOS 客户端 / Android 客户端 简体中文 / 正體中文 / English

**OAuth2 解决了什么问题?**  
任何的身份认证, 本质上都是  
基于对请求方的不信任产生的

# 理解 OAuth2 协议



# 协议流程和角色

- 资源所有者 ( resource owner )
- 客户端 ( client )
- 资源服务器 ( resource server )
- 授权服务器 ( authorization server )

[RFC 6749](#)

[QQ oAuth](#)

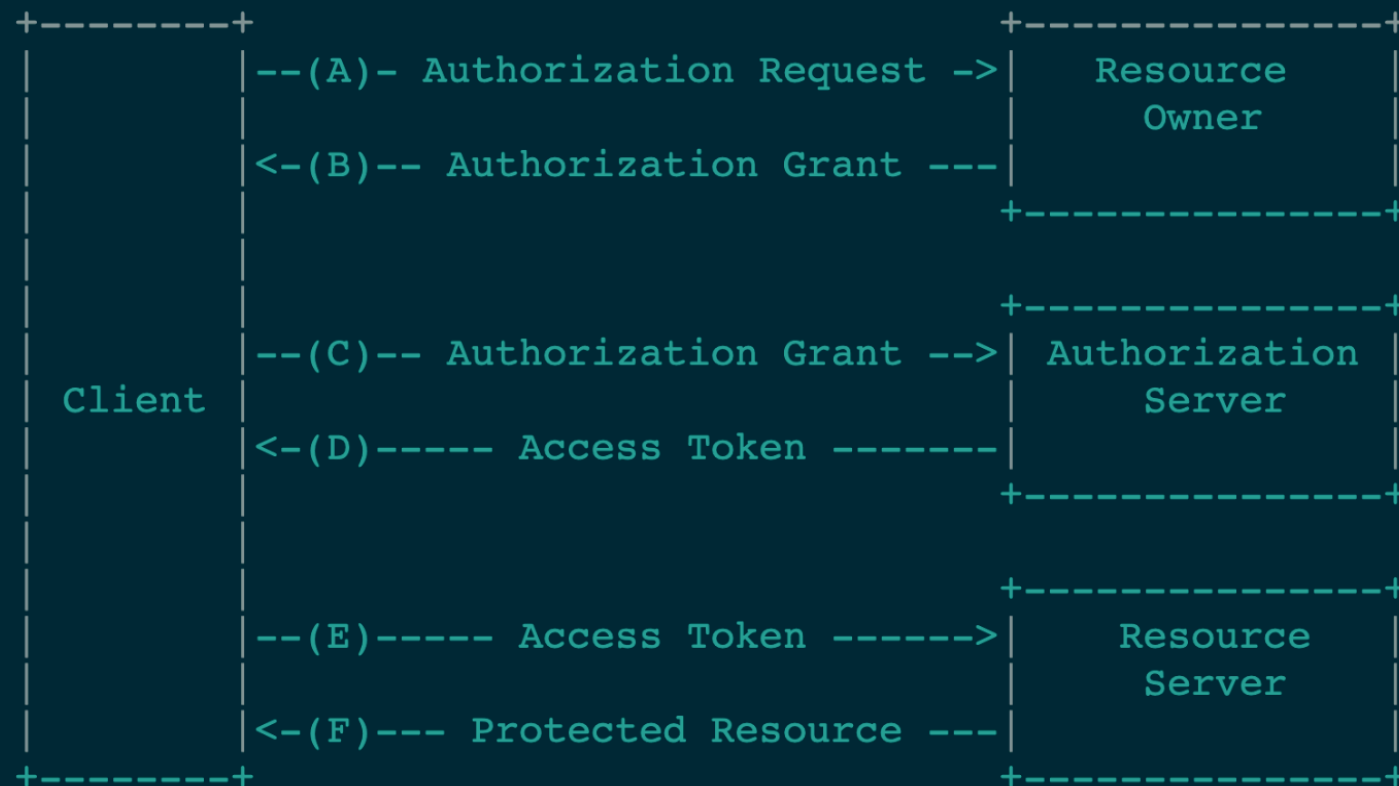


Figure 1: Abstract Protocol Flow

# OAuth2 授权所需信息

- 应用名称
- 应用网站
- 重定向 URI 或回调 URL ( redirect\_uri )
- 客户端标识 client\_id
- 客户端密钥 client\_secret

[oauth2 授权](#)

## Register a new OAuth application

Application name \*

Something users will recognize and trust.

Homepage URL \*

The full URL to your application homepage.

Application description

This is displayed to all users of your application.

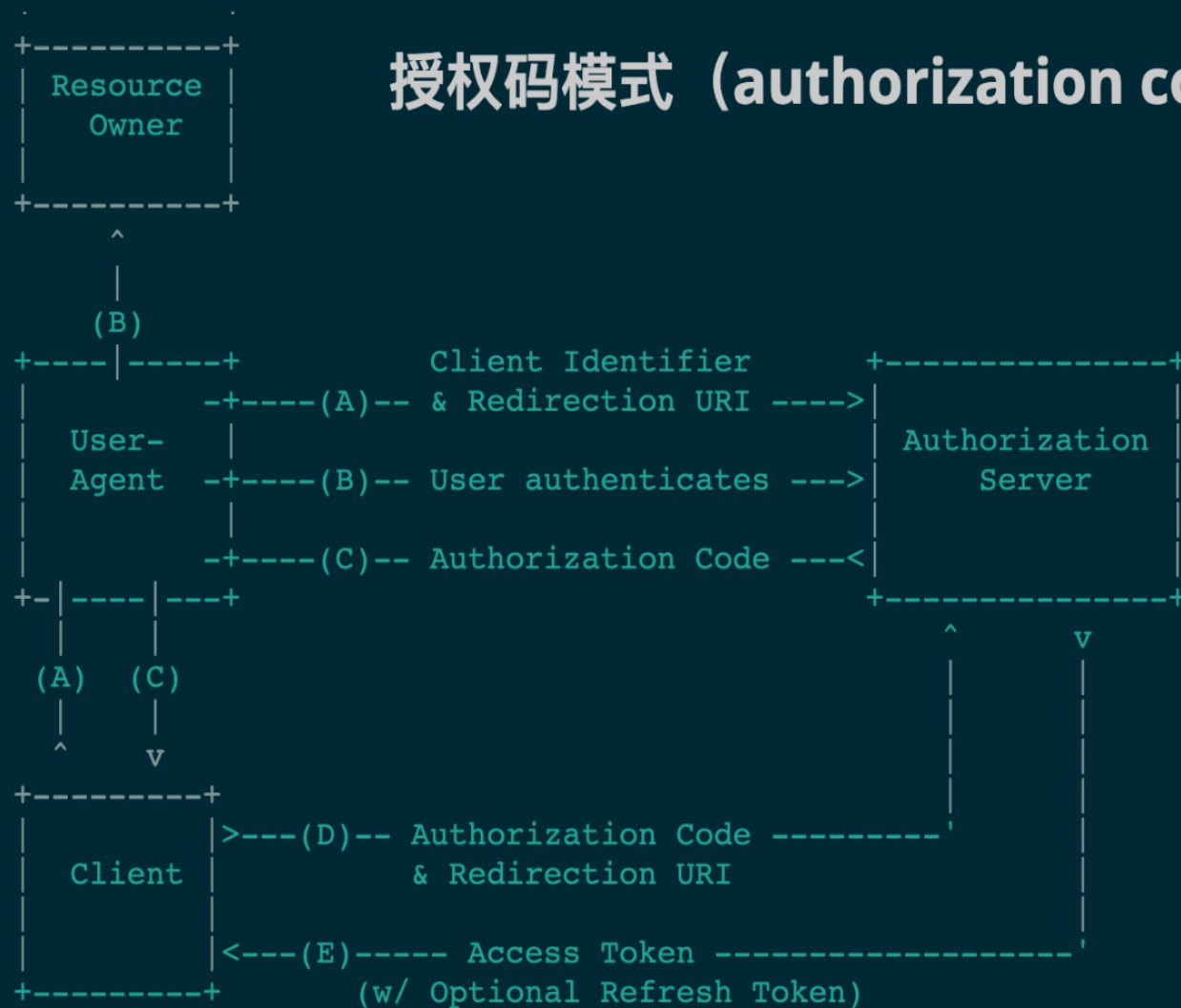
Authorization callback URL \*

Your application's callback URL. Read our [OAuth documentation](#) for more information.

Register application

[Cancel](#)

## 授权码模式 (authorization code)



Ruby-China在后台通过code + client\_id + client\_secret获取AccessToken



# 授权码模式 所需参数

```
HTTP/1.1 200 OK
```

```
Content-Type: application/json;charset=UTF-8
```

```
Cache-Control: no-store
```

```
Pragma: no-cache
```

```
{  
  "access_token": "2YotnFZFEjrlzCsicMWpAA",  
  "token_type": "example",  
  "expires_in": 3600,  
  "refresh_token": "tGzv3JOkF0XG5Qx2TlKWIA",  
  "example_parameter": "example_value"  
}
```

## 简化模式 (implicit grant)

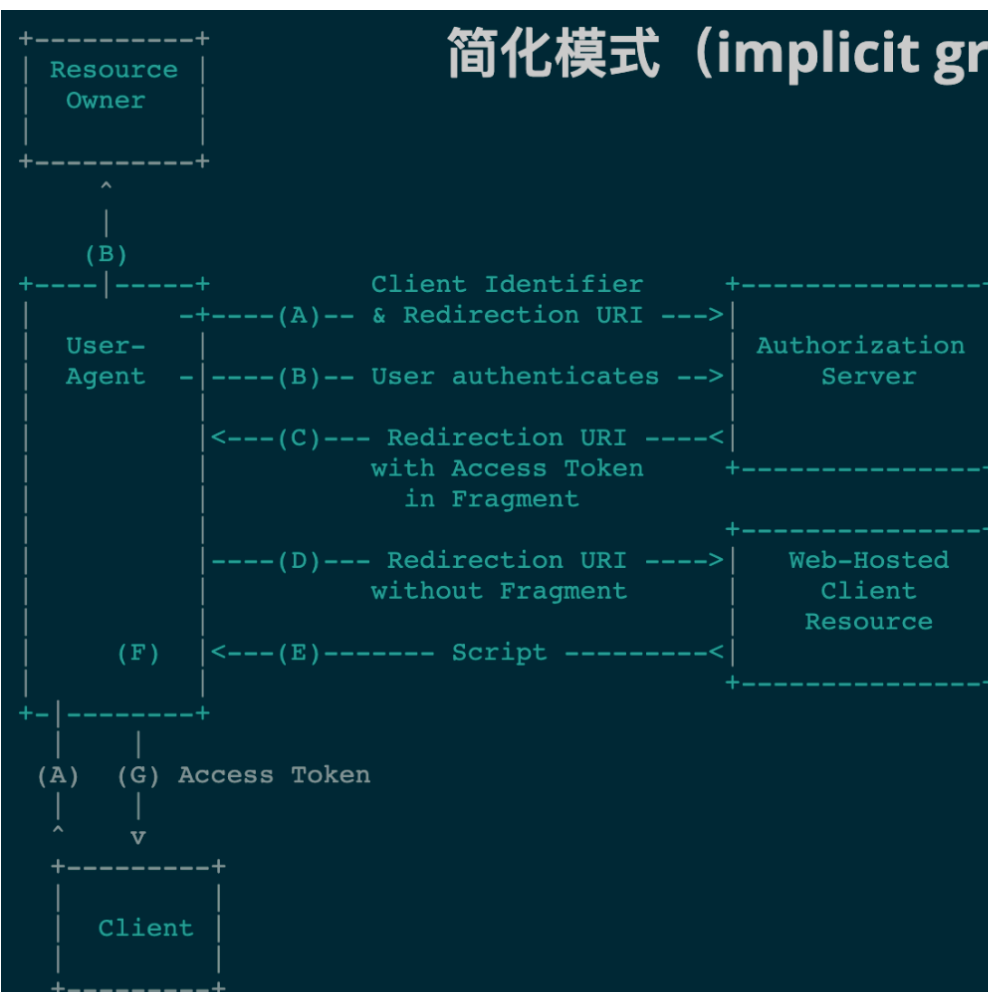
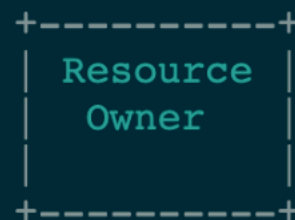


Figure 4: Implicit Grant Flow

ount/sign in



v  
|  
Resource Owner  
(A) Password Credentials  
|  
v



>--(B)---- Resource Owner ----->  
Password Credentials  
  
<--(C)---- Access Token -----<  
(w/ Optional Refresh Token)



## 密码模式

(Resource Owner Password Credentials Grant)

Figure 5: Resource Owner Password Credentials Flow

## 客户端模式 ( Client Credentials Grant )

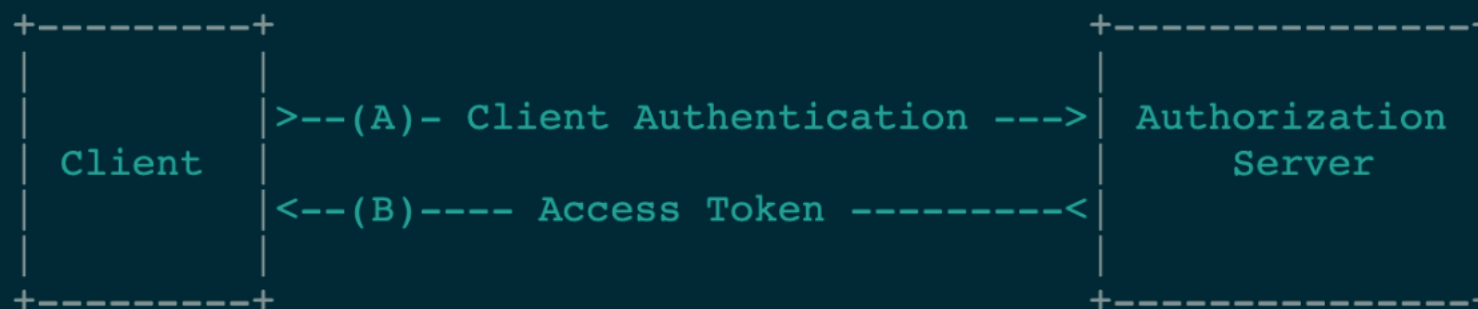


Figure 6: Client Credentials Flow

## OAuth2 授权方式小结

- 授权码模式：正宗的Oauth认证，推荐
- 密码模式：客户端为自家公司的应用，为遗留项目设计
- 简化模式：为没有后台的项目设计
- 客户端模式：一般用户后端API的相关操作

# 单点登录

概念：在多个系统中，用户只需登录一次，各个系统即可感知该用户已经登录。



# 单点登录实现方案

## 1.同父域下的单点登录解决方案

如[hr.oa.com],[km.oa.com],[fuli.oa.com]

cookie domain属性共享cookie，session共享

基于cookie开源项目代表：JWT

## 2.不同域下的单点登录解决方案

如[www.taobao.com],[www.tmall.com]

基于中央认证服务器开源项目代表：CAS

# 主流常用的登录方案

## 1.同一公司，同父域下的单点登录解决方案

如[hr.oa.com],[km.oa.com],[fuli.oa.com]

基于cookie开源项目代表：JWT

## 2.同一公司，不同域下的单点登录解决方案

如[www.taobao.com],[www.tmall.com]

基于中央认证服务器开源项目代表：CAS

## 3.不同公司，不同域下的 第三方登录功能实现

如第三方网站支持微信登录，QQ登录，微博登录等

基于 oAuth2.0 协议各大公司自己的支持



# THANKS