**统一认证服务对接文档**

**V1.0**

|  |  |  |
| --- | --- | --- |
| 版本 | 修改人 | 内容 |
| V1.0 | 邹龙明 | 初稿 |
|  |  |  |
|  |  |  |

广州市品高软件股份有限公司

2018年7月

# 本文目的

帮助需要对接品高统一认证的第三方应用技术人员快速接入统一认证服务。

# 阅读对象

项目经理，技术经理，开发人员。

# 服务地址

## 测试环境

待定

## 正式环境

待定

# 名词解释

## 基本认证(Basic Authentication)

基本认证(Basic Authentication)是指在一次HTTP请求中传递参数用于验证本次请求的客户端身份的验证方式。

在品高统一认证服务中，**一般用于验证应用身**份。

包含基本认证的HTTP请求需要添加如下请求头：

Authorization: Basic ${Base64.encode(clientId+":"+clientSecret)}

示例：

clientId="s6BhdRkqt3";

clientSecret="gX1fBat3bV";

请求如下：

POST /oauth2/token

Host: sso.example.com

Authorization: Basic czZCaGRSa3F0MzpnWDFmQmF0M2JW

更详细信息请参考[RFC2617](https://tools.ietf.org/html/rfc2617)

## 访问令牌（access token）

访问令牌(Access Token)，简称at，是OAuth2中定义的一个用于标识用户和应用身份的令牌，由统一认证服务器生成和颁发给应用。

### 访问令牌的身份信息

访问令牌代表的身份有三种可能：

* 只代表用户身份
* 只代表应用身份
* 代表用户和应用身份

在品高统一认证服务中，**不允许不受信任的应用获取访问令牌**，因此只会有两种情况：

* 只代表应用身份
* 代表用户和应用身份

### 访问令牌的类型

品高统一认证服务支持两种类型的访问令牌：

* UUID
* JWT

UUID是统一认证服务生成的一个全局唯一的字符串，如：

01369670-d390-4da1-8580-9eb95af81fac

JWT是一个经过签名的字符串，如：

eyJhbGciOiJSUzI1NiIsImtpZCI6IjFlOWdkazcifQ.ewogImlzcyI6ICJodHRwOi8vc2VydmVyLmV4YW1wbGUuY29tIiwKICJzdWIiOiAiMjQ4Mjg5NzYxMDAxIiwKICJhdWQiOiAiczZCaGRSa3F0MyIsCiAibm9uY2UiOiAibi0wUzZfV3pBMk1qIiwKICJleHAiOiAxMzExMjgxOTcwLAogImlhdCI6IDEzMTEyODA5NzAKfQ.ggW8hZ1EuVLuxNuuIJKX\_V8a\_OMXzR0EHR9R6jgdqrOOF4daGU96Sr\_P6qJp6IcmD3HP99Obi1PRs-cwh3LO-p146waJ8IhehcwL7F09JdijmBqkvPeB2T9CJNqeGpe-gccMg4vfKjkM8FcGvnzZUN4\_KSP0aAp1tOJ1zZwgjxqGByKHiOtX7TpdQyHE5lcMiKPXfEIQILVq0pc\_E2DzL7emopWoaoZTF\_m0\_N0YzFC6g6EJbOEoRoSK5hoDalrcvRYLSrQAZZKflyuVCyixEoV9GfNQC3\_osjzw2PAithfubEEBLuVVk4XUVrWOLrLl0nx7RkKU8NXNHq-rvKMzqg

关于JWT的更多说明请看[JWT(JSON Web Token)](#_JWT（JSON_Web_Token）)章节

### 访问令牌安全

在品高统一认证服务中，访问令牌是服务端认证客户端身份的充分条件，因此访问令牌不可泄露和共享，以免被其他应用伪造身份发送请求。

如果需要共享访问令牌，建议重新申请一个没有任何权限的普通应用，使用这个应用来申请访问令牌，并将访问令牌共享给其他应用

## 登录令牌(Login Token)

登录令牌(Login Token)，简称lt，是品高统一认证在OpenID Connect的基础上拓展的协议。

登录令牌可以作为用户身份认证的凭据，即允许用户不输入用户名和密码的情况下使用登录令牌登录。

### 使用场景

通过lt，可以让用户在使用桌面应用打开Web应用时，不用再输入用户名和密码就能自动登录。

### 使用流程

使用lt的过程如下：

* 获取lt
* 通过浏览器调用统一认证服务的登录接口，并将lt附带在请求中
* 统一认证服务认证lt完成，重定向回到web应用

### 有效时间

每个lt的有效时间是5分钟，并且只能使用一次。

## JWT（JSON Web Token）

JWT(JSON Web Token)是为了在网络应用环境间传递声明而执行的一种基于JSON的开放标准。

### JWT是什么

一般是一个经过签名的字符串，通过**.**分割为三个部分，如：eyJhbGciOiJSUzI1NiIsImtpZCI6IjFlOWdkazcifQ**.**ewogImlzcyI6ICJodHRwOi8vc2VydmVyLmV4YW1wbGUuY29tIiwKICJzdWIiOiAiMjQ4Mjg5NzYxMDAxIiwKICJhdWQiOiAiczZCaGRSa3F0MyIsCiAibm9uY2UiOiAibi0wUzZfV3pBMk1qIiwKICJleHAiOiAxMzExMjgxOTcwLAogImlhdCI6IDEzMTEyODA5NzAKfQ**.**ggW8hZ1EuVLuxNuuIJKX\_V8a\_OMXzR0EHR9R6jgdqrOOF4daGU96Sr\_P6qJp6IcmD3HP99Obi1PRs-cwh3LO-p146waJ8IhehcwL7F09JdijmBqkvPeB2T9CJNqeGpe-gccMg4vfKjkM8FcGvnzZUN4\_KSP0aAp1tOJ1zZwgjxqGByKHiOtX7TpdQyHE5lcMiKPXfEIQILVq0pc\_E2DzL7emopWoaoZTF\_m0\_N0YzFC6g6EJbOEoRoSK5hoDalrcvRYLSrQAZZKflyuVCyixEoV9GfNQC3\_osjzw2PAithfubEEBLuVVk4XUVrWOLrLl0nx7RkKU8NXNHq-rvKMzqg

第一部分称为header，第二部分称为playload，第三部分称为signature。

header和playload是两个base64编码的json字符串，直接用base64解码即可，以上面的jwt为例，解码的结果如下：



signature是签名信息，JWT是否有效取决于对签名信息的验证是否通过。

### 品高统一认证的JWT

在品高统一认证服务中，一般只会颁发两种JWT：

* 访问令牌的JWT
* ID Token的JWT

关于id token请看*[OpenID Connect Section 2](http://openid.net/specs/openid-connect-core-1_0.html" \l "IDToken)*

访问令牌使用RS256签名算法生成signature

ID Token使用HS256算法生成signature

### playload的编码规范

由于jwt是需要在网络中传递的字符串，因此在base64编码时，会对playload进行url安全的编码，即：

* 去除尾部的"="
* 把"+"替换成"-"
* 把"/"替换成"\_"

在解码playload时，需要先按照如下步骤进行url解码：

* 把"-"替换成"+"
* 把"\_"替换成"/"
* (计算BASE64URL编码长度)%4

a. 结果为0，不做处理

b. 结果为2，字符串添加"=="

c. 结果为3，字符串添加"="

url安全的解码完成之后，再进行base64解码，即可得到最初编码时的json。

## 认证服务器

即统一认证服务本身

## 资源服务器

对外提供业务服务，需要认证服务器认证客户端身份

## 客户端

通过认证服务器登录和调用资源服务器的应用

## Client Id

第三方应用的唯一标识，在【网关控制台】->【我的客户端】里面注册时，由系统自动分配。

## Client Secret

第三方应用的鉴权使用的密钥，在【网关控制台】->【我的客户端】里面注册时，由系统自动分配。

## 授权码

授权码是系统用以表示用户登录成功的一个字符串，获取授权码可以用于授权客户端登录。可以通过登录接口或者是专门的授权码获取接口获取。

## ID token

ID Token是一个使用HS256算法和第三方应用的Client Secret作为密钥签名的JWT，第三方应用必须使用自己的Client Secret验证ID Token的有效性，ID token中用户信息都存储在payload中，具体字段如下表。

| **属性** | **类型** | **说明** |
| --- | --- | --- |
| sub | string | *REQUIRED*，标准属性，值是用户ID，代表当前登录的用户 |
| aud | string | *REQUIRED*，标准属性，值是客户端的[ClientId](http://dev.bingocc.com/ua/GLOSSARY.html#clientid) |
| exp | long | *REQUIRED*，标准属性，代表这个ID Token的过期时间， 值是距离标准时间1970-01-01T00:00:00Z的**秒数** |
| username | string | *REQUIRED*，拓展属性，用户登录账号 |
| name | string | *OPTIONAL*，拓展属性，用户名 |
| login\_name | string | *DEPRECATED*，*OPTIONAL*，拓展属性，用户登录账号，已经废弃使用，请使用**username** |

# 功能介绍

品高统一认证服务是一个基于OpenID Connect协议的单点登录和统一认证服务。

## 相关协议

本节简单介绍品高统一认证服务支持和涉及的相关协议。

### OpenID Connect

OpenID Connect是一套基于OAuth 2.0协议的轻量级规范,提供通过API进行身份交互的框架。

品高统一认证服务实现了OpenID Connect协议，并在该协议的基础上增加了更多的企业特性。

更多关于OpenID Connect的详情请参考[OpenID Connect Core](http://openid.net/specs/openid-connect-core-1_0.html)

### OAuth2.0

OAuth2.0是一个用户授权认证协议，关注客户端开发者的简易性，同时为Web应用，桌面应用和手机，和起居室设备提供专门的认证流程。

关于OAuth2的协议详情请看*[RFC6749](https://tools.ietf.org/html/rfc6749)*

品高统一认证服务支持OAuth2协议，允许应用使用OAuth2的方式接入用户身份认证。

### LDAP

LDAP是轻量目录访问协议，它是基于X.500标准的，但是简单多了并且可以根据需要定制。与X.500不同，LDAP支持TCP/IP，这对访问Internet是必须的。

关于LDAP协议的详情请参考[RFC4511](https://tools.ietf.org/html/rfc4511)

品高统一认证服务目前实现了LDAP的两个操作：

* BindRequest
* SearchRequest

只支持客户端应用使用LDAP协议接入登录。

## 基础功能

品高统一认证服务提供了如下基础功能：

1. 单点登录
2. 统一身份认证（用户+应用）
3. 获取用户详细信息
4. 获取应用详细信息

## 单点登录

在多个应用系统间，实现一次登录，就可以自动登录全部应用系统，不需要用户在多个应用中输入用户名和密码。

## 统一身份认证

所有的客户端请求都由认证服务器统一认证用户身份，所有的资源服务器都可以从认证服务器获取到本次请求的客户端身份和用户身份。

## 获取用户详细信息

资源服务器可以通过认证服务器获取用户更详细的信息，如邮箱，生日等。

## 获取应用详细信息

资源服务器可以通过认证服务器获取客户端应用的详细信息，如应用名称，应用负责人等。

## 高级功能

品高统一认证服务提供了更多企业级的高级功能

* 代理认证
* JWT颁发

## 代理认证

品高统一认证服务支持受信任的第三方应用代理用户和应用认证过程。

## JWT颁发

品高统一认证服务支持颁发JWT作为访问令牌。

# 接入方案

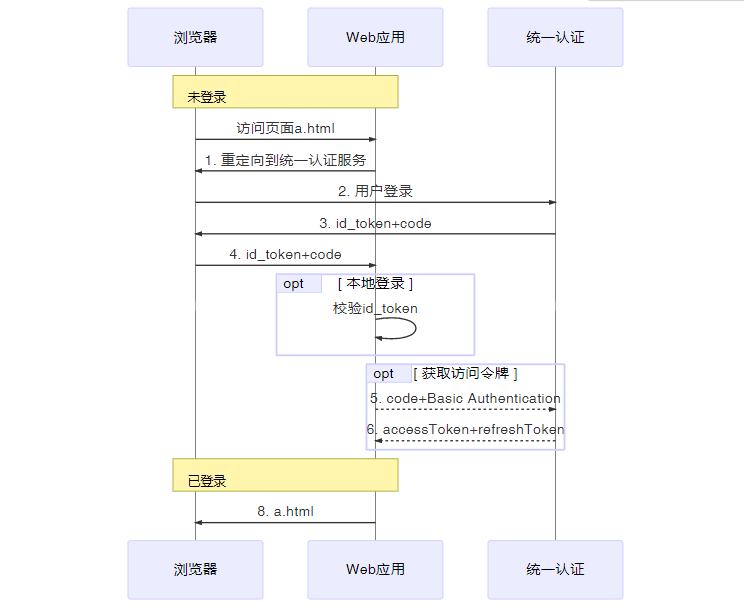
## 场景1：web页面接入

WEB页面接入的方案主要适用于在电脑端通过浏览器访问的系统应用，解决用户在浏览器打开多个web应用是需要输入多次用户名密码登录的问题，实现用户一次登录，多个系统使用的效果。

Web页面集成统一认证的原理是在通过在浏览器打开集成了统一认证的某应用的页面时，页面重定向跳转到统一认证登录界面，登录之后页面重新回到用户请求的目标页面，而此时，如果用户在相同浏览器再打开另外一个集成了统一认证的页面时，由于该页面也会跳转到统一认证登录页，因为是在相同浏览器，统一认证登录的session仍然有效，所以会自动验证用户已经登录，不需要重新登录，便可以自动跳转到用户访问真正的目标页面。

### Web登录过程

Web页面接入的登录流程如下：



1. 当用户首次访问Web应用时，Web应用会将请求重定向到统一认证的登录页面,重定向的过程一般是重定向到[登录接口](#_登录接口_2)，这个过程重定向需要传三个参数：

* client\_id
* response\_type
* redirect\_uri

具体的参数含义请查阅7.1章节登录接口。

注意：web应用可以直接在页面通过前端JS，也可以通过后台返回302的方式让浏览器重定向到统一认证的登录接口。

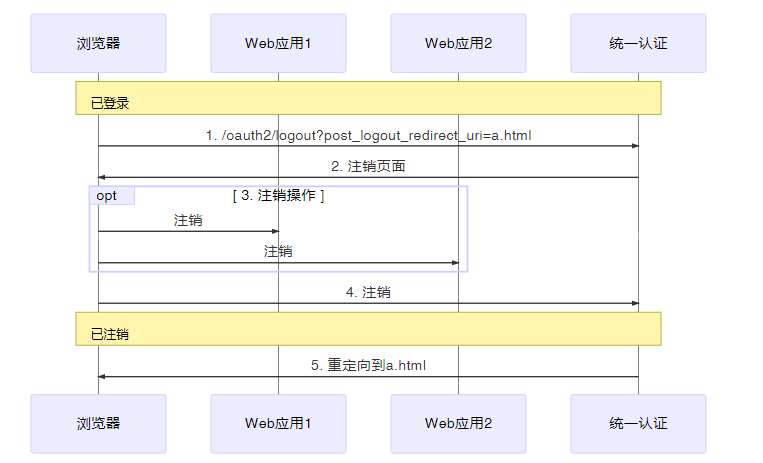
1. 用户在登录页面输入用户名密码后提交登录,这个过程与Web应用无关，由统一认证服务管控。
2. 统一认证服务会颁发授权码（code）和id\_token给Web应用，并将用户重定向到Web应用。

注意：id\_token是一个JWT，使用client\_id对应的client\_secret签名，Web应用使用自己的clientSecret对id\_token进行验证，通过后即可进行用户本地登录。

1. Web应用通过id\_token验证用户身份，完成本地登录。验证id\_token之后，从id\_token的playload中即可获取用户信息，playload中的字段含义参考名词解释4.11章节。
2. 如果有需要访问接入了统一认证的其他应用的接口，或者是发布在品高API Gateway的API接口时，可以通过授权码(code)+ 基本认证获取访问令牌(access\_token,简称at)。如果需要为这个用户获取at的话，可以通过授权码获取访问令牌，获取后自行保存。
3. 统一认证返回访问令牌给Web应用保存。访问令牌的详细信息请看[获取访问令牌](#_获取访问令牌_1)

### Web注销过程

Web注销流程如下：



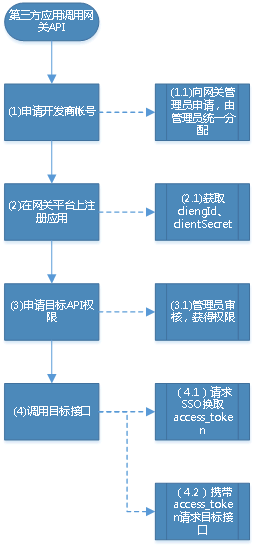
* + - 1. 这里的注销地址在浏览器直接重定向即可，接口说明请看[注销接口](#_注销接口_1)。

1. ua会在注销接口中直接返回一个注销页面，这个页面中包含了所有已登录的应用的注销地址。
2. 注销所有已登录的应用
3. 注销统一认证服务
4. 重定向到最初传给注销接口的url。

注：在整个注销过程中，实际上需要客户端做的操作只是重定向到注销接口，其他的流程都是统一认证服务配合浏览器自动完成

## 场景2：调用网关API

该场景描述目标接口是私有的（需要鉴权）的情况下，第三方应用服务需要预先准备的步骤，如果目标接口公开的（无需鉴权），那么直接调用即可，无需描述。



### 调用API网关示意图



### 申请开发商开发帐号

第三方应用开发产商如需调用网关API，首先需要有一个网关的帐号，一般由网关管理员统一分配，如无帐号，可主动联系管理员，让管理员分配一个开发商帐号，使用该帐号可在网关上发布API，注册应用。

### 在网关平台上注册应用

只有在网关平台上注册的应用，才有权限去调用网关API接口（如果API的使用控制是公开的，可以跳过此步，一般建议给每个调用系统在我的应用中注册一个客户端，以方便接入统一的单点登录服务）

应用申请流程：（更详细申请流程可参考API控制台操作手册）

进入网关控制台，在“我的客户端”中为调用系统注册一个客户端，注册成功后，系统会返回该应用的clientId和[clientSecret](#_Client Secret)，后面API网关、单点登录服务都会根据clientId和clientSecret唯一标识。

### 申请目标API权限

如果要调用的API为私有API，则应用在使用前申请对应API的权限，开发者可以进入授权API的Tab页面，申请该API的权限，API权限申请功能，可能在不同部署环境会有所不同，比如：需要审批等。待审批后，第三方应用拥有调用目标API的权限，即可以调用目标API。

### 调用目标接口

应用调用目标接口时，需要在请求头中添加一个键值对

Authorization: Bearer ${access token}

所以，应用需要使用步骤6.4.2中获取的clientId和clientSecret、开发商的账户、开发商的密码去SSO换取access token，再拼接请求头去调用目标接口。

向SSO换取access token方式：

1）使用SDK进行获取，使用该种方式需要接入SSO SDK

AccessToken token = client.obtainAccessTokenByClientCredentials();// client 是配置应用信息节点中的 client

2）使用标准RESTful接口进行获取

接口地址

POST /oauth2/token

参数说明

grant\_type REQUIRED, 授权方式，通过请求体传递，在使用用户名密码的方式下固定为password。

username REQUIRED，用户登录名，通过请求体传递。

password REQUIRED，用户登录密码，通过请求体传递。

Authorization。[REQUIRED](http://dev.bingocc.com/ua/GLOSSARY.html" \l "required" \o "必需参数)，在请求头中传递，使用基本认证的方式验证客户端身份,  
如Authorization: Basic ${Base64.encode(clientId+":"+clientSecret)}

请求示例

POST /oauth2/token HTTP/1.1

Host: sso.example.com

Content-Type: application/x-www-form-urlencoded

Authorization: Basic czZCaGRSa3F0MzpnWDFmQmF0M2JW

username=zhangs&passowrd=123456

正确响应：HTTP/1.1 200 OK

{

"access\_token": "52f96681-fc47-4415-bdb7-4d8fc5136c46",

"token\_type": "bearer",

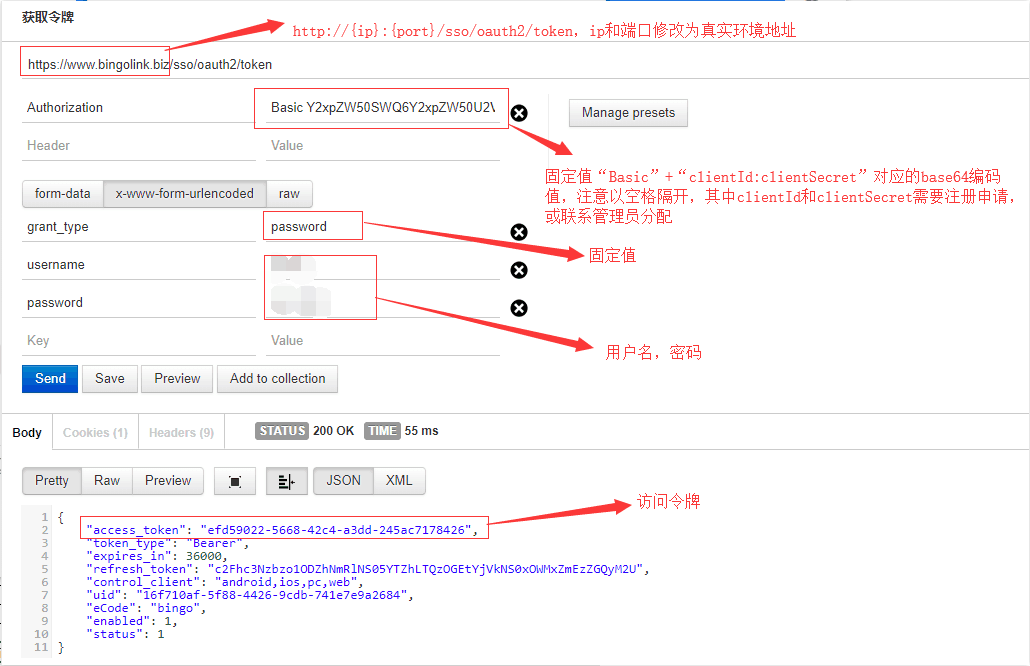
"expires\_in": 36000,

"refresh\_token": "e9a9ac6d-9454-4517-b78c-22215a37dbba"

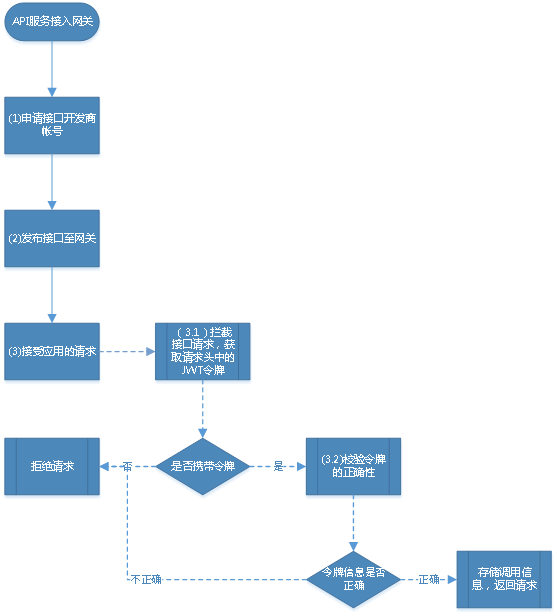
}

在成功响应中已经返回**access\_token**参数，再使用该access\_token向目标接口发送请求即可。

以下是模拟向SSO获取access token的请求



## 场景3：发布API到网关



### 使用网关管理页面发布接口

#### 注册应用开发商账户

第三方应用开发产商在网关平台上发布API，首先需要有一个网关的帐号，一般由网关管理员统一分配，如无帐号，可主动联系管理员，让管理员分配一个开发商帐号，使用该帐号可在网关上发布API，注册应用。（某些场景下可能是管理员直接代替开发商发布接口，那么可以省略该步骤，直接由管理员发布接口即可）

#### 向API网关注册接口

API开发完成后，开发人员需要将API部署到某台服务器上，要发布到API网关上，要求该服务器能被网关所在的服务器访问到。

API部署成功后，开发人员可以登录API网关控制台，创建API，维护API介绍信息、及导入该API的接口信息，详细操作流程如下：（更详细申请流程可参考API控制台操作手册，另外如果是由管理员发布接口，那么该过程可以忽略）

1）创建API

登录网关控制台，进入“我的API”模块，点击创建API，录入API的基本信息。

2）导入API接口

方式一：直接通过界面的方式，一个接口一个接口的创建，创建接口定义时，需先在页面的右边添加一个Restful的路径信息，添加完Path后，才能在Path上添加操作。

方式二：导入一份JSON格式API定义，该API定义需满足OpenAPI 2.0规范要求，可以直接以JSON方式导入，也可以通过Url的方式导入。

3）维护API文档

为了方便API的调用者对API的功能，及使用有更深入的了解，API提供者可以在控制台上创建一些API相关的文档，来介绍自己的API，具体可以点击“文档管理”菜单，进入API文档管理界面，开发 人员可以通过Markdown的语法来编辑API的文档。

4）API的发布

API创建成功，并维护好该API的接口及文档信息后，可以进入“发布部署”菜单，将该API发布到对应的网关上，发布成功后，就可以通过网关地址访问该API了。

* 1. 管理员审核

在一些场景下，需要管理员审核API，管理员通过后才能允许被使用，此时管理员需要审核各大厂商发布的API。（某些场景下可能是管理员直接代替开发商发布接口，那么可以省略该步骤，直接由管理员发布接口即可）

### 网关接受已注册应用的请求

应用根据6.4中描述的方式对API进行请求，如果目标接口使用的是**私有（需要鉴权）**的方式，那么此时，网关将会首先拦截应用服务器发送过来的流量。

网关会根据应用传递的access\_token判定调用方应用是否已经授权，合法，验证通过后，会把应用信息加密成为[JWT](#_JWT（JSON Web Token）)，并添加到请求头中，再把流量转发至API服务器。如果不合法，那么网关将拒绝该请求，响应状态码为401的响应。

如果该接口配置成公开（无需鉴权）的方式，那么：

网关将会直接透传请求至API服务器。

**注意：此时请求头中如包含Authorization参数，网关将会主动进行认证，认证成功，将会把应用信息或账户信息加密成为[JWT](#_JWT（JSON Web Token）)并传递给API服务器，认证失败依旧透传（请求参数原样传输）请求至API服务器。**

API服务需要拦截请求，获取请求头中的JWT信息

Authorization: Bearer ${JWT token}

### 接口服务校验JWT token

如果接口权限是私有的，那么网关将在请求头中传递Authorization $jwt token给API服务器。

JWT token（详见4.4） 是由网关成功校验后传递过来的含有用户或应用身份信息一串令牌，其中header和playload是两个base64编码的json字符串，直接用base64解码即可。将获取应用的基本信息。而另一部分，signature是签名信息，JWT是否有效取决于对签名信息的验证是否通过。signature解密使用的是RS256签名算法。此时，通过以下地址，获取SSO公钥信息。

接口说明

GET /oauth2/publickey

注：旧版使用如下接口，该接口已废弃，目前仍然可用，但可能在未来的版本删除。

GET /publickey

接口示例

GET /oauth2/publickey

Host: sso.example.com

返回结果：

HTTP/1.1 200 OK

MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDDASOjIWexLpnXiJNJF2pL6NzP

fBoF0tKEr2ttAkJ/7f3uUHhj2NIhQ01Wu9OjHfXjCvQSXMWqqc1+O9G1UwB2Xslb

WNwEZFMwmQdP5VleGbJLR3wOl3IzdggkxBJ1Q9rXUlVtslK/CsMtkwkQEg0eZDH1

VeJXqKBlEhsNckYIGQIDAQAB

得到公钥后，对signature进行解密，与header.playload进行加密后的数据进行比对，校验JWT是否被串改，以此判断此时的请求是否是合法，合法即正确响应，不合法则抛出异常响应。

当成功校验JWT token后，即可以使用base64对playload部分进行解码，解码之后，将得到用户的一些相关信息如下所示：

| **属性** | **类型** | **说明** |
| --- | --- | --- |
| sub | string | [*REQUIRED*](http://dev.bingocc.com/ua/GLOSSARY.html#required)，标准属性，用户ID |
| aud | string | [*OPTIONAL*](http://dev.bingocc.com/ua/GLOSSARY.html#optional), 标准属性，**access\_token**所代表的[ClientId](http://dev.bingocc.com/ua/GLOSSARY.html" \l "clientid" \o "应用唯一标识，注册应用的时候填写)，只有当**access\_token**包含了应用身份才会有这个属性 |
| exp | long | [*REQUIRED*](http://dev.bingocc.com/ua/GLOSSARY.html#required)，标准属性，代表这个[JWT](http://dev.bingocc.com/ua/def/jwt.html)过期时间， 值是距离标准时间1970-01-01T00:00:00Z的**秒数** |
| username | string | [*REQUIRED*](http://dev.bingocc.com/ua/GLOSSARY.html#required)，拓展属性，用户登录账号 |
| scope | string | [*REQUIRED*](http://dev.bingocc.com/ua/GLOSSARY.html#required)，标准属性，授权列表，即**access\_token**中包含的授权列表 |
| user\_id | string | [*DEPRECATED*](http://dev.bingocc.com/ua/GLOSSARY.html#deprecated)，*[REQUIRED](http://dev.bingocc.com/ua/GLOSSARY.html" \l "required" \o "必需参数)*，拓展属性，同**sub**属性 |
| client\_id | string | [*DEPRECATED*](http://dev.bingocc.com/ua/GLOSSARY.html#deprecated)，*[OPTIONAL](http://dev.bingocc.com/ua/GLOSSARY.html" \l "optional" \o "可选参数)*，拓展属性，同**aud**属性 |
| 其他属性 | - | [*OPTIONAL*](http://dev.bingocc.com/ua/GLOSSARY.html#optional)，可能存在其他拓展插件添加的字段 |

除此之外，如果API接口不使用网关进行鉴权，那么也可通过类似OAuth2模式进行权限控制管理。如在请求头中使用

Authorization: Bearer ${access token}

拦截获取后与SSO交互进行鉴权。

# 接口说明

## 登录接口

登录接口单点登录的入口，所有接入统一认证的Web应用，都需要重定向到这个接口引导用户登录。

这个认证接口是单点登录的统一入口，允许通过不同的参数决定服务端不同的认证行为。

|  |  |
| --- | --- |
| URL | http://serverIp:port/serverPath/oauth2/authorize |
| 调用方式 | GET |
| 参数 | redirect\_uri:  REQUIRED，在queryString或请求体中传递，需要先进行编码，这个参数用于决定用户登录完成后的重定向地址，同时服务端会校验这个地址是否符合client\_id参数对应的应用允许的回调地址。 如果client\_id所代表的应用不允许回调到这个地址，这个认证请求会被中断。  response\_type：  REQUIRED，在queryString或请求体中传递，这个参数是指定统一认证服务回调到redirect\_uri这个地址的时候，需要附加的用户登录信息，为一个枚举类型，可选值如下：  code：授权码，会在回调请求中添加code参数  code id\_token：授权码+ID Token，会在回调请求中添加code参和id\_token参数  client\_id：  REQUIRED，在queryString或请求体中传递，这个参数是应用唯一标识，表名用户需要在哪个应用上登录。  x-authz-url：  OPTIONAL，这参数在请求头中传递，值是一个URL，表示本次认证请求希望由指定URL的服务器代理认证，如果请求中添加了这个请求头，统一认证服务会直接将请求重定向到这个URL，不做所有认证处理。  login\_token：  OPTIONAL，登录令牌，在queryString或请求体中传递，值是一个字符串，表示本次认证请求通过login\_token参数直接确认用户身份，不需要打开用户登录页面。 |
| 返回值 | 登录页面 |
| 属性说明 | 用户通过登录页面输入用户名和密码登录成功后，统一认证服务会将授权码和ID Token作为参数附加在**redirect\_uri**后面并重定向到这个地址。 客户端可以通过ID Token获取用户信息，完成本地登录  ID Token  ID Token是一个使用HS256算法和ClientSecret作为密钥签名的JWT，客户端必须使用自己的ClientSecret验证ID Token的有效性。 |
| 备注 | 该接口用于根据ID获取租户信息。 |

## 注销接口

单点登录完成之后，当用户需要注销时，必须通过单点注销注销所有已经登录的应用系统。

单点注销需要完成以下两步：

* 注销所有已登录的系统
* 注销统一认证

|  |  |
| --- | --- |
| URL | http://serverIp:port/serverPath/oauth2/logout |
| 调用方式 | GET |
| 参数 | post\_logout\_redirect\_uri  OPTIONAL，注销完成后跳转地址，没有传的话默认会跳转到统一认证根目录。 |
| 示例 | GET /oauth2/logout?  post\_logout\_redirect\_uri=http%3A%2F%2Fwww.bingosoft.net HTTP/1.1  Host: sso.example.com |

## 获取访问令牌

访问令牌是OAuth 2.0的一个核心概念，作为认证客户端身份的充分条件。

访问令牌的生成和颁发都由认证服务器处理。

**接口说明**

在品高统一认证服务中，获取访问令牌有多种不同的方式，适用于不同的场景

1. 通过用户名和密码

只有被授权允许获取用户密码的应用才允许使用这种方式获取at。

1. 通过授权码

绝大多数的Web应用使用这种方式获取at，获取的过程如下：

* 用户访问Web应用
* 应用重定向到统一认证服务的登录页
* 用户在统一认证服务的页面中输入用户名密码
* 统一认证服务验证用户身份，并颁发授权码
* 统一认证服务重定向到Web应用并传递授权码
* Web应用通过授权码获取at

在这个过程中，Web应用只能拿到授权码，不会导致用户密码泄露。

1. 通过ClientSecret

某些服务不关系用户身份，只关心应用身份（如：定时任务），这种情况下可以使用应用资深的ClientSecret获取at(仅代表应用身份，参考访问令牌(Access Token))， 使用这个at来调用这些服务即可。

1. 通过at和ClientSecret

有些时候【服务A】需要调用【服务B】来完成自己的业务操作，由于【服务B】只对【服务A】开放权限，并且需要用户身份。

那么当【服务A】接收到客户端请求时，就必须使用代表【服务A】和客户端用户身份的at才能调用【服务B】，这种情况下就需要使用at和ClientSecret来获取新的at，以便调用【服务B】

注： 统一认证服务规范要求，资源服务器之间，不使用客户端请求中带过来的at进行相互调用，因此当我们的服务需要调用其他服务时，需要使用这种方式获取一个代表资源服务器自身的at。

1. 通过刷新令牌

at获取后，会有一个过期时间，at过期后需要重新获取，但是为了让用户不用重新输入用户名和密码，可以使用这种方式刷新at。

注: 刷新at不是延迟at的过期时间，而是重新生成at，旧的at依然会过期。

### 用户名密码生成

|  |  |
| --- | --- |
| URL | http://serverIp:port/serverPath/oauth2/token |
| 调用方式 | GET |
| 参数 | grant\_type  REQUIRED, 授权方式，通过请求体传递，在使用用户名密码的方式下固定为password。  username  REQUIRED，用户登录名，通过请求体传递。  password  REQUIRED，用户登录密码，通过请求体传递。  Authorization  REQUIRED，在请求头中传递，使用[基本认证](#_基本认证(Basic_Authentication))的方式验证客户端身份, |
| 返回值 | {  "access\_token": "52f96681-fc47-4415-bdb7-4d8fc5136c46",  "token\_type": "bearer",  "expires\_in": 36000,  "refresh\_token": "e9a9ac6d-9454-4517-b78c-22215a37dbba"  } |
| 说明 | "access\_token": [*REQUIRED*](http://dev.bingocc.com/ua/GLOSSARY.html#required)，标准属性，访问令牌,  "token\_type":REQUIRED，标准属性，令牌类型，目前固定为bearer,  "expires\_in"REQUIRED，标准属性，过期时间，表明从返回这个请求的时间开始计算，到这个令牌过期的秒数:,  "refresh\_token":REQUIRED，标准属性，刷新令牌，  其他属性：可能存在其他拓展属性 |

### 授权码生成

通过授权码获取at，是绝大多数不被授权允许使用用户密码的应用使用的方式。授权码的获取可以参考登录接口和获取授权码。

|  |  |
| --- | --- |
| URL | http://serverIp:port/serverPath/oauth2/token |
| 调用方式 | GET |
| 参数 | grant\_type  REQUIRED, 授权方式，通过请求体传递，在使用授权码的方式下固定为authorization\_code。  code  REQUIRED，授权码，通过请求体传递。  Authorization  REQUIRED，在请求头中传递，使用[基本认证](#_基本认证(Basic_Authentication)_7)的方式验证客户端身份,  如Authorization: Basic czZCaGRSa3F0MzpnWDFmQmF0M2JW  id\_token  OPTIONAL，通过请求体传递，表明是否需要在返回结果中增加id\_token属性，默认为false，如果需要在返回结果中增加id\_token，则传true。 |
| 返回值 | {  "access\_token": "52f96681-fc47-4415-bdb7-4d8fc5136c46",  "token\_type": "bearer",  "expires\_in": 36000,  "refresh\_token": "e9a9ac6d-9454-4517-b78c-22215a37dbba"  } |
| 说明 | "access\_token": [*REQUIRED*](http://dev.bingocc.com/ua/GLOSSARY.html#required)，标准属性，访问令牌,  "token\_type":REQUIRED，标准属性，令牌类型，目前固定为bearer,  "expires\_in"REQUIRED，标准属性，过期时间，表明从返回这个请求的时间开始计算，到这个令牌过期的秒数:,  "refresh\_token":REQUIRED，标准属性，刷新令牌，  其他属性：可能存在其他拓展属性 |

### ClientSecret生成

通过ClientSecret生成at，这个at仅代表应用身份，只有当需要调用的服务不关注用户，只关注应用的场景下才需要这样的at。

|  |  |
| --- | --- |
| URL | http://serverIp:port/serverPath/oauth2/token |
| 调用方式 | GET |
| 参数 | grant\_type  REQUIRED, 授权方式，通过请求体传递，在使用ClientSecret的方式下固定为client\_credentials。  Authorization  REQUIRED，在请求头中传递，使用[基本认证](#_基本认证(Basic_Authentication)_1)的方式验证客户端身份, |
| 返回值 | {  "access\_token": "52f96681-fc47-4415-bdb7-4d8fc5136c46",  "token\_type": "bearer",  "expires\_in": 36000,  "refresh\_token": "e9a9ac6d-9454-4517-b78c-22215a37dbba"  } |
| 说明 | "access\_token": [*REQUIRED*](http://dev.bingocc.com/ua/GLOSSARY.html#required)，标准属性，访问令牌,  "token\_type":REQUIRED，标准属性，令牌类型，目前固定为bearer,  "expires\_in"REQUIRED，标准属性，过期时间，表明从返回这个请求的时间开始计算，到这个令牌过期的秒数:,  "refresh\_token":REQUIRED，标准属性，刷新令牌，  其他属性：可能存在其他拓展属性 |

### at和secret换取

品高统一认证服务规范要求，资源服务器之间，不使用客户端请求中带过来的at进行相互调用。

因此当我们的服务需要调用其他服务时，需要使用这种方式获取一个代表资源服务器自身的at。

|  |  |
| --- | --- |
| URL | http://serverIp:port/serverPath/oauth2/token |
| 调用方式 | GET |
| 参数 | grant\_type  REQUIRED, 授权方式，通过请求体传递，在使用ClientSecret的方式下固定为client\_credentials。  access\_token  REQUIRED，代表客户端用户的访问令牌，通过请求体传递。  Authorization  REQUIRED，在请求头中传递，使用[基本认证](#_基本认证(Basic_Authentication)_2)的方式验证客户端身份, |
| 返回值 | {  "access\_token": "52f96681-fc47-4415-bdb7-4d8fc5136c46",  "token\_type": "bearer",  "expires\_in": 36000,  "refresh\_token": "e9a9ac6d-9454-4517-b78c-22215a37dbba"  } |
| 说明 | "access\_token": [*REQUIRED*](http://dev.bingocc.com/ua/GLOSSARY.html#required)，标准属性，访问令牌,  "token\_type":REQUIRED，标准属性，令牌类型，目前固定为bearer,  "expires\_in"REQUIRED，标准属性，过期时间，表明从返回这个请求的时间开始计算，到这个令牌过期的秒数:,  "refresh\_token":REQUIRED，标准属性，刷新令牌，  其他属性：可能存在其他拓展属性 |

### 刷新访问令牌

第一次获取at之后，经过一定的时间at就会过期，过期后需要刷新访问令牌才能继续使用，否则就需要用户重新输入用户名密码进行授权了。（刷新完成之后，会生成新的at和rt，旧的at和rt都会失效）

|  |  |
| --- | --- |
| URL | http://serverIp:port/serverPath/ /oauth2/token |
| 调用方式 | GET |
| 参数 | grant\_type  REQUIRED, 授权方式，通过请求体传递，在使用ClientSecret的方式下固定为client\_credentials。  refresh\_token  REQUIRED，刷新令牌，通过请求体传递。  Authorization  REQUIRED，在请求头中传递，使用[基本认证](#_基本认证(Basic_Authentication)_3)的方式验证客户端身份, |
| 返回值 | {  "access\_token": "52f96681-fc47-4415-bdb7-4d8fc5136c46",  "token\_type": "bearer",  "expires\_in": 36000,  "refresh\_token": "e9a9ac6d-9454-4517-b78c-22215a37dbba"  } |
| 说明 | "access\_token": [*REQUIRED*](http://dev.bingocc.com/ua/GLOSSARY.html#required)，标准属性，访问令牌,  "token\_type":REQUIRED，标准属性，令牌类型，目前固定为bearer,  "expires\_in"REQUIRED，标准属性，过期时间，表明从返回这个请求的时间开始计算，到这个令牌过期的秒数:,  "refresh\_token":REQUIRED，标准属性，刷新令牌，  其他属性：可能存在其他拓展属性 |

## 获取授权码

获取授权码可以用于授权客户端登录。在OpenID Connect中， 授权码通过授权码流程获取，在品高统一认证中的实现可以查看登录接口。

除了按照标准协议使用登录接口获取授权码外，品高统一认证也提供了更加方便第三方应用作为代理认证服务器的授权码获取接口。

|  |  |
| --- | --- |
| URL | http://serverIp:port/serverPath/oauth2/authzcode |
| 调用方式 | POST |
| 参数 | client\_id  REQUIRED，需要授权的ClientId，统一认证服务会为这个应用生成授权码。  access\_token  REQUIRED，用于认证第三方授权应用身份的at，不需要用户身份，服务端会根据at做如下处理：   * 解析应用身份 * 判断应用是否有权限做代理认证 * 通过认证后颁发授权码 |
| 返回值 | {  "code": "eb0ea764a30f43d6b8a75fa7f682bd11",  "expires\_in": 300  } |
| 说明 | code：REQUIRED，授权码  expires\_in: REQUIRED，有效时间，指的是从接收到这个响应开始计算，到授权码过期的秒数; |

## 获取登录令牌

登录令牌是一次性认证的工具，可以用于从桌面应用打开Web应用实现单点登录。

|  |  |
| --- | --- |
| URL | http://serverIp:port/serverPath/oauth2/logintoken |
| 调用方式 | GET |
| 参数 | access\_token  REQUIRED，访问令牌，通过queryString或请求体传递，用于确认用户身份，这里的at必须同时代表用户和应用身份。  Authorization  REQUIRED，通过请求头传递，使用[基本认证](#_基本认证(Basic_Authentication)_8)验证应用身份，这里的应用身份必须和access\_token代表的应用身份一致。 |
| 返回值 | {  "login\_token":"b0ed4b75-6bc5-4005-8d04-47168fb8c50d",  "expires\_in": 60  } |
| 说明 | login\_token: REQUIRED，登录令牌;  expires\_in: REQUIRED，过期时间，指的是从收到这个响应后开始，到lt过期的秒数，默认60秒 |

## 访问令牌签名

使用已经颁发的at签名生成JWT的接口，接口需要使用基本认证验证客户端身份，客户端必须向管理员申请获得使用这个接口的权限。

|  |  |
| --- | --- |
| URL | http://serverIp:port/serverPath/oauth2/jwt |
| 调用方式 | GET |
| 参数 | access\_token  REQUIRED，访问令牌，在queryString或请求体中传递。  Authorization  REQUIRED，在请求头中传递，使用[基本认证](#_基本认证(Basic_Authentication)_4)验证客户端身份，如果access\_token所代表的应用身份与请求头中的应用身份不一致，则拒绝颁发JWT。  resource  OPTIONAL，通过queryString或请求体传递，值是ClientId，用于获取access\_token对指定ClientId的权限列表(scope)， 没有指定的情况下，会使用Authorization中验证的ClientId作为resource。  alg  OPTIONAL，签名算法，在queryString或请求体中传递，固定为rsa。  这里颁发的JWT可以作为at使用，资源服务器需要从统一认证服务器获取签名公钥对这个JWT验证，获取公钥的接口请看*获取签名公钥* |
| 返回值 | {  "jwt\_token":"eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiJ9.eyJjbGllbnRfaWQiOiJzNkJoZFJrcXQzIiwidXNlcl9pZCI6IjQzRkU2NDc2LUNEN0ItNDkzQi04MDQ0LUM3RTMxNDlEMDg3NiIsInVzZXJuYW1lIjoiYWRtaW4iLCJzY29wZSI6Imp3dCIsImV4cCI6MTQ5OTM4OTAyNX0.bCjDJFZzECgH0WeAS2LsQF0MyHBRQuCCijye78zty9YSHNRmDJopB1dEecwFcufRvbhKbgLnLLmab-FAR3suvLA8pHEv1msy-x4\_6wzgbFO8oIcljjNv5a5SEcX59YtrEzTtytBPlVPySnsHmqsiUQwLjE3fQTiQ3dX2sLb99h8"  } |
| 说明 | jwt\_token: REQUIRED，根据请求参数颁发的JWT对象;  这个签发的JWT包含了at所代表用用户信息和应用信息，这些信息存储在playload中，属性如下：  sub:REQUIRED，标准属性，用户ID  aud:OPTIONAL, 标准属性，access\_token所代表的ClientId，只有当access\_token包含了应用身份才会有这个属性  exp:REQUIRED，标准属性，代表这个JWT过期时间， 值是距离标准时间1970-01-01T00:00:00Z的秒数  username:REQUIRED，拓展属性，用户登录账号  scope:REQUIRED，标准属性，授权列表，即access\_token中包含的授权列表  user\_id:DEPRECATED，REQUIRED，拓展属性，同sub属性  client\_id:DEPRECATED，OPTIONAL，拓展属性，同aud属性  其他属性:OPTIONAL，可能存在其他拓展插件添加的字段 |

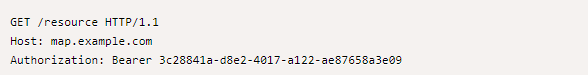
## 验证访问令牌

资源服务器在接收请求的时候，可以通过请求头中(或请求参数中)的at验证客户端身份。

客户端传递at支持两种方式：

C:\Users\pengyl\AppData\Roaming\Tencent\Users\602147250\TIM\WinTemp\RichOle\4~I@WYM2JR}CW~%JDQWZ[$I.png

或



资源服务器需要支持两种at格式：

* JWT
* UUID

### 验证JWT格式的at

JWT格式的at，一般是RS256算法签名的JWT，验证的时候需要获取签名公钥进行验证，公钥获取请看获取签名公钥。

获取公钥验证通过后，可以将playload进行BASE64解码，得到一个JSON对象。

JSON对象的属性说明参考访问*[令牌签名](#_访问令牌签名)*。

### 验证UUID格式的at

UUID格式的at是一个无业务意义的字符串，验证这个字符串需要通过授权服务器。

验证接口：

|  |  |
| --- | --- |
| URL | http://serverIp:port/serverPath/oauth2/tokeninfo |
| 调用方式 | POST |
| 参数 | Authorization  REQUIRED，通过请求头传递，使用[基本认证](#_基本认证(Basic_Authentication)_9)验证客户端身份，如：Authorization: Basic czZCaGRSa3F0MzpnWDFmQmF0M2JW，这里的[基本认证](#_基本认证(Basic_Authentication)_5)验证的是资源服务器的身份。  access\_token  REQUIRED，通过queryString或请求体传递，如access\_token=3c28841a-d8e2-4017-a122-ae87658a3e09  resource  OPTIONAL，通过queryString或请求体传递，值是ClientId，用于获取access\_token对指定ClientId的权限列表(scope)， 没有指定的情况下，会使用Authorization中验证的ClientId作为resource。 |
| 返回值 | {  "user\_id": "43FE6476-CD7B-493B-8044-C7E3149D0876",  "username": "admin",  "expires\_in": 23634,  "client\_id": "link",  "scope": "read,write"  } |
| 说明 | {  "user\_id": REQUIRED，用户ID,  "username": REQUIRED，用户登录账号,  "expires\_in":REQUIRED，at过期时间，值是指从收到这个返回的时间开始到过期的秒数,  "client\_id": OPTIONAL，ClientId，如果at没有应用信息则不返回这个属性,  "scope": OPTIONAL，授权列表，没有任何授权的情况下不返回这个属性} |

## 获取用户信息

统一认证服务颁发at后，资源服务器可以使用at用于认证用户身份，如果需要获取用户更详细的信息，需要使用获取用户信息的接口。

|  |  |
| --- | --- |
| URL | http://serverIp:port/serverPath/oauth2/userinfo |
| 调用方式 | GET |
| 参数 | access\_token  REQUIRED，访问令牌，通过queryString或请求体传递，用于确认用户身份，这里的at必须同时代表用户和应用身份。  Authorization  REQUIRED，通过请求头传递，使用[基本认证](#_基本认证(Basic_Authentication)_6)验证应用身份，这里的应用身份必须和access\_token代表的应用身份一致。 |
| 返回值 | {  "sub": "0400cfcf-6b29-4191-a7af-17e50924864e",  "name": "张三",  "gender": "male",  "email": "zhangs@bingosoft.net",  "phone\_number": "12345678901",  "username": "zhans"  } |
| 说明 | {  "sub": REQUIRED，标准属性，表示用户ID,  "name": REQUIRED，标准属性，表示用户全名,  "gender": OPTIONAL，标准属性，性别，female表示女性，male表示男性,  "email": OPTIONAL，标准属性，电子邮箱地址,  "phone\_number": OPTIONAL，标准属性，联系电话,  "username": REQUIRED，拓展属性，表示用户登录账号,  “org\_id”: OPTIONAL，拓展属性，用户所在的部门id  } |

## 获取应用信息

统一认证服务颁发at后，资源服务器可以使用at用于认证客户端应用身份，如果需要获取客户端应用更详细的信息，需要使用获取应用信息的接口。

获取应用信息在OpenID Connect中无标准定义，这是品高统一认证服务提供的扩展接口。

|  |  |
| --- | --- |
| URL | http://serverIp:port/serverPath/oauth2/clientinfo |
| 调用方式 | GET |
| 参数 | access\_token  REQUIRED，访问令牌，通过queryString或请求体传递，用于确认用户身份，这里的at必须同时代表用户和应用身份。  Authorization  REQUIRED，通过请求头传递，使用[基本认证](#_基本认证(Basic_Authentication)_10)验证应用身份，这里的应用身份必须和access\_token代表的应用身份一致。 |
| 返回值 | {  "aud": "s6BhdRkqt3",  "name": "地图服务",  "enabled": "true"  "redirect\_uri": "https://map.example.com",  "redirect\_uri\_pattern": "https://map.example.com/login/\*\*",  "logout\_uri": "https://map.example.com/logout"  } |
| 说明 | {  "aud": REQUIRED，表示ClientId,  "name": REQUIRED，表示应用的名称,  "enabled": OPTIONAL，表示应用是否被禁言，true表示启用，false表示禁用  "redirect\_uri":OPTIONAL，表示应用回调地址,  "logout\_uri": OPTIONAL，表示应用的注销地址  } |

## 获取签名公钥

品高统一认证服务支持为at签名生成JWT，参考访问令牌签名。

这里签名默认使用RS256算法，因此资源服务器验证JWT需要使用公钥验证。

统一认证服务提供了公钥的获取接口。。

|  |  |
| --- | --- |
| URL | http://serverIp:port/serverPath/oauth2/publickey |
| 调用方式 | GET |
| 参数 |  |
| 返回值 | MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDDASOjIWexLpnXiJNJF2pL6NzPfBoF0tKEr2ttAkJ/7f3uUHhj2NIhQ01Wu9OjHfXjCvQSXMWqqc1+O9G1UwB2XslbWNwEZFMwmQdP5VleGbJLR3wOl3IzdggkxBJ1Q9rXUlVtslK/CsMtkwkQEg0eZDH1VeJXqKBlEhsNckYIGQIDAQAB |
| 说明 |  |

## 刷新登录会话

多个应用系统接入统一认证登录后，如果统一认证的登录会话过期，此时会导致某些未登录的应用在登录会话过期后需要重新输入用户名和密码登录。

为了保证统一认证的会话不过期，可以使用这个接口刷新登录会话。

|  |  |
| --- | --- |
| URL | http://serverIp:port/serverPath/oauth2/session\_refresh |
| 调用方式 | GET |
| 参数 | auth\_token\_${contextPath}，在cookie中传递，这里的${contextPath}由统一认证服务部署的上下文决定，如果部署在域名根目录下，则为root，如：  auth\_token\_root  auth\_token\_ssov3  这个cookie在统一认证中登录完成后，统一认证服务会在登录完成的响应中写这个cookie，如果是使用web页面刷新的话，通过隐藏的iframe直接调用这个地址即可，浏览器会自动带上cookie。如果是通过其他用户代理，请自己从登录过程中解析出cookie并自行刷新。 |
| 返回值 | {  Set-Cookie: auth\_token\_ssov3=eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJuYW1lIjoiYWRtaW4iLCJleHAiOjE1MDA0NzgwMzUsImp0aSI6ImE0ZTE1ZWRlLWJkNDAtNDFhMi1iNmE5LWQ3ZDNjNTUzZmIyMiJ9.cGeRopGfWMxNHo\_zoToFOrlWS-hbEBZVgRLLZ5r6cW0;  } |
| 说明 |  |

## 创建登录会话

这个接口是基于OAuth2协议的授权接口（品高统一认证已实现，参考登录接口）扩展的接口，需要在标准定义的参数的基础上，增加几个参数。

这个接口会为浏览器创建一个session，并且返回一个json对象，客户端可以按照json对象设置浏览器对应的cookie或者请求头等参数，即相当于已经登录， 重定向到统一认证服务时，统一认证服务会自动认证用户身份，不会打开登录页面。

使用这个接口需要较高权限，普通应用没有权限使用，需要使用权限请向管理员申请。

|  |  |
| --- | --- |
| URL | http://serverIp:port/serverPath/oauth2/authorize |
| 调用方式 | GET |
| 参数 | auth\_token\_${contextPath}，在cookie中传递，这里的${contextPath}由统一认证服务部署的上下文决定，如果部署在域名根目录下，则为root，如：  auth\_token\_root  auth\_token\_ssov3  这个cookie在统一认证中登录完成后，统一认证服务会在登录完成的响应中写这个cookie，如果是使用web页面刷新的话，通过隐藏的iframe直接调用这个地址即可，浏览器会自动带上cookie。如果是通过其他用户代理，请自己从登录过程中解析出cookie并自行刷新。 |
| 返回值 | {  auth\_token\_ssov3=eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJuYW1lIjoiYWRtaW4iLCJleHAiOjE1MDA0NzgwMzUsImp0aSI6ImE0ZTE1ZWRlLWJkNDAtNDFhMi1iNmE5LWQ3ZDNjNTUzZmIyMiJ9.cGeRopGfWMxNHo\_zoToFOrlWS-hbEBZVgRLLZ5r6cW0;  } |
| 说明 |  |

## 应用管理

应用管理功能，是OAuth2的扩展协议，主要是扩展OAuth2中的client动态注册和管理的协议。

协议参考文档包含三篇：

* [OAuth 2.0 Dynamic Client Registration Protocol[RFC7591]](https://tools.ietf.org/html/rfc7591)：OAuth2扩展协议，主要定义Client的动态注册接口。
* [OpenID Connect Dynamic Client Registration 1.0](http://openid.net/specs/openid-connect-registration-1_0.html)：OIDC协议扩展定义，包含应用注册和查询接口定义。
* [OAuth 2.0 Dynamic Client Registration Management Protocol[RFC7592]](https://tools.ietf.org/html/rfc7592)：OAuth2扩展协议，主要定义Client的读改删接口。

品高SSO目前只支持[RFC7591](https://tools.ietf.org/html/rfc7591" \t "http://dev.bingocc.com/ua/protocol/_blank)扩展协议，另外两个协议会在后续的版本逐步提供。

### 注册应用

应用动态注册接口，可以用于应用动态注册，参考协议RFC7591。

|  |  |
| --- | --- |
| URL | http://serverIp:port/serverPath/oauth2/register |
| 调用方式 | GET |
| 参数 | Authorization  REQUIRED，在请求头中传递，需要传递AccessToken，如：Authorization: Bearer {access\_token}。  Client  REQUIRED，在请求体中传递，可以为空的json，如{},不能不传，也可以是标准文档定义的属性，参考RFC7591。 |

标准协议中定义的所有属性都是可选的，在品高统一认证服务中，扩展了几个特定的属性：

属性 类型 说明

client\_id string OPTIONAL，应用ID，不指定的情况下会自动生成

client\_secret string OPTIONAL，应用密钥，不指定的情况下会自动生成

响应结果属性说明：

属性 类型 说明

client\_id string 应用ID

client\_secret string 应用密钥

client\_id\_issued\_at long 应用创建时间，表示距离标准时间1970-01-01T00:00:00Z的秒数

client\_secret\_expires\_at long 应用密钥过期时间，表示距离标准时间1970-01-01T00:00:00Z的秒数，0表示不过期

e\_code string 租户id，默认同请求的accessToken中代表的用户的ECode

# 下载地址

**统一认证接口示例互联网地址**

http://dev.bingocc.com/ua/