# 统一认证系统实施简要说明

## *版本说明*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 内容 | 日期 | 作者 |  |
| 1 | 建立初始版本 | 2016-12-29 | Riki |  |
| 2 | 增加Role的相关内容；增加用例、安全、代办相关内容和说明；增加移动端的使用说明。 | 2016-12-30 | Riki |  |

## *备注*

统一认证系统基于OpenID和OAuth实现。与Openid相关的概念遵从Openid Connect 1.0规范；与OAuth相关的概念遵从OAuth2.0规范。

## Openid vs Oauth

OpenID与Oauth其实没有很严格的区分，关键在于你把用户基本信息怎么定义。OpenID只管认证（用户名、密码登录）以及需要时获取用户基本信息不能用于访问其它资源。而Oauth是认证之后还多了一套访问资源的流程。

OpenID Connect在OAuth 2.0授权流程的基础上，扩展实现了认证功能。

### OpenID的流程

OpenID定义了3个认证流程

1. 授权码模式（Authorization Code Flow）
2. 简化模式（Implicit Flow）
3. 混合模式（Hybrid Flow）

此外还需支持

1. 认证后获取用户信息的流程

### OAuth的流程

OAuth定义了4个认证流程

1. 授权码模式（Authorization code Flow）
2. 简化模式（Implicit Flow）
3. 密码模式（Resource Owner Password Credentials Flow）
4. 客户端模式（Client Credentials Flow）

此外还需支持令牌更新

1. 更新令牌流程

统一认证系统需要支持以上流程。其中对于授权码模式和简化模式，OpenID与OAuth基本一致。

### OpenID对OAuth的扩展

OpenID相关的流程中，客户端发起的请求扩展了范围(scope)值类型，增加了openid类型。认证执行返回的信息除了OAuth要求的access token等数据项外，增加了一个名叫 ID Token的数据，格式为JWT。

* ID Token内的Claims数据

ID Token中的claims包含了用于用户认证的信息。

ID\_Token的claims字段：

iss：必选项。认证提供方（组织）信息。是一个https的url。

sub：必选项。认证提供者（组织下的人的标识）信息。不能超过255个ASCII码。

aud：必选项。认证信息的使用方。必须包含客户端的client\_id。

exp：必选项。过期时间信息（1970-01-01 0:0:0之后的秒数）

iat：必选项。认证提供的时间点。

auth\_time：如果认证请求参数包含max\_age或scope中要求auth\_time，那么是必选的。

nonce：用于将客户端session和ID\_Token关联起来，防范重放攻击。此值在用户请求验证码的参数中提供给认证服务器，在ID\_Token中必须原样保留。

* Scope对于id\_token内Claims数据的影响（常见约定）

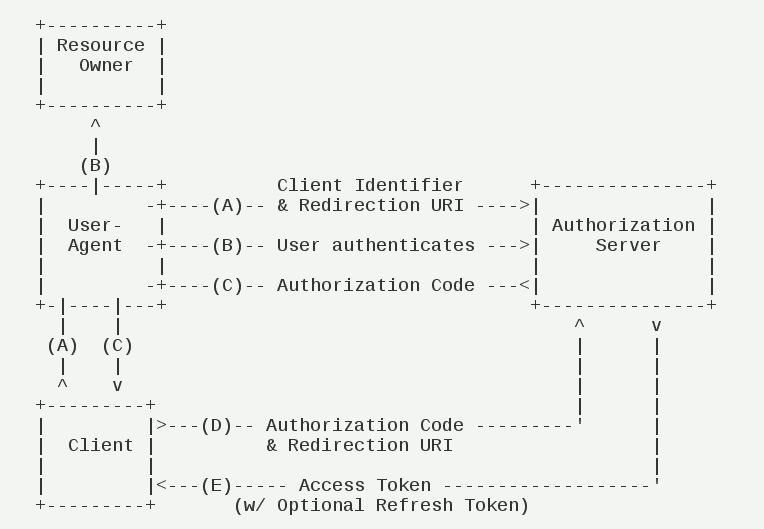
如果scope是openid，那么id\_token只包含iss、sub、aud、exp和iat

如果scope是openid email，那么id\_token还要包含email、email\_verified

## 认证各流程概要与输入输出

*说明：响应数据如果不是以302重定向形式在URI中携带，那么通常以json格式返回。*

### 授权码模式（Authorization Code flow）



（A）用户访问客户端，后者将前者导向认证服务器。

（B）用户选择是否给予客户端授权。

（C）假设用户给予授权，认证服务器将用户导向客户端事先指定的"重定向URI"（redirection URI），同时附上一个授权码。

（D）客户端收到授权码，附上早先的"重定向URI"，向认证服务器申请令牌。这一步是在客户端的后台的服务器上完成的，对用户不可见。

（E）认证服务器核对了授权码和重定向URI，确认无误后，向客户端发送访问令牌（access token）和更新令牌（refresh token）。

此流程主要分为两个阶段。

阶段1，申请授权码

输入（即客户端请求，访问地址为Authorization Endpoint，以上步骤A）

* response\_type：表示授权类型，必选项，此处的值固定为"code"
* 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，如需执行openid流程则必须包含openid
* state：表示客户端的当前状态，可以指定任意值，认证服务器必须原样返回。

输出（即服务端响应，以上步骤C）：此处响应为前一步指定的redirect\_uri及对应参数。

* code：表示授权码，必选项。该码的有效期应该很短，通常设为10分钟，客户端只能使用该码一次，否则会被授权服务器拒绝。该码与客户端ID和重定向URI，是一一对应关系。
* state：如果客户端的请求中包含这个参数，认证服务器的回应必须原样返回。

阶段2，用授权码换取Token

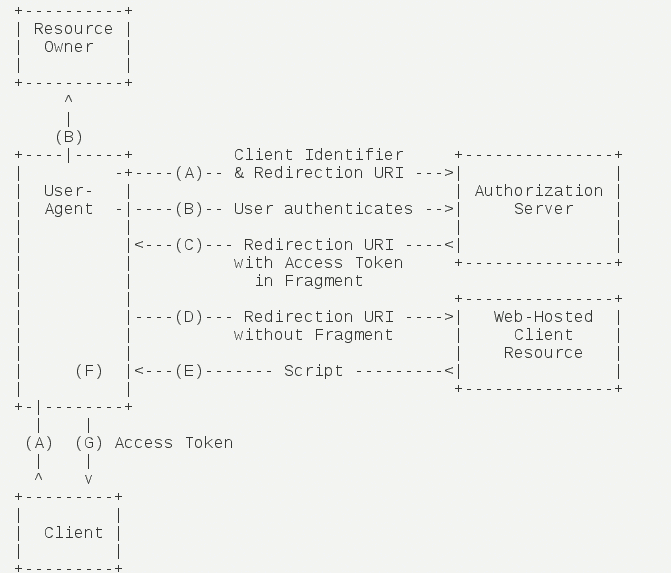
输入（即客户端请求，访问地址为Token Endpoint，以上步骤D）

* 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A步骤中的该参数值保持一致。
* client\_id：表示客户端ID，必选项。

输出（即服务端响应，以上步骤E）

* access\_token：表示访问令牌，必选项。
* token\_type：表示访问令牌类型，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。
* id\_token：含有claim信息的JWT。必选项。

### 简化模式（Implicit Flow）



（A）客户端将用户导向认证服务器。

（B）用户决定是否给于客户端授权。

（C）假设用户给予授权，认证服务器将用户导向客户端指定的"重定向URI"，并在URI的Hash部分包含了访问令牌。（即图中的Fragment，也就是URL中通常在#后面的部分）

（D）浏览器向资源服务器发出请求，其中不包括上一步收到的Hash值。

（E）资源服务器返回一个网页（脚本），其中包含的代码可以获取Hash值中的Token。

（F）浏览器执行上一步获得的脚本，提取出Token。

（G）浏览器将令牌发给客户端。

此流程只需要一个阶段即可直接获得所有令牌。令牌对访问者是可见的，客户端不需要认证。且此流程禁止发放Refresh Token。

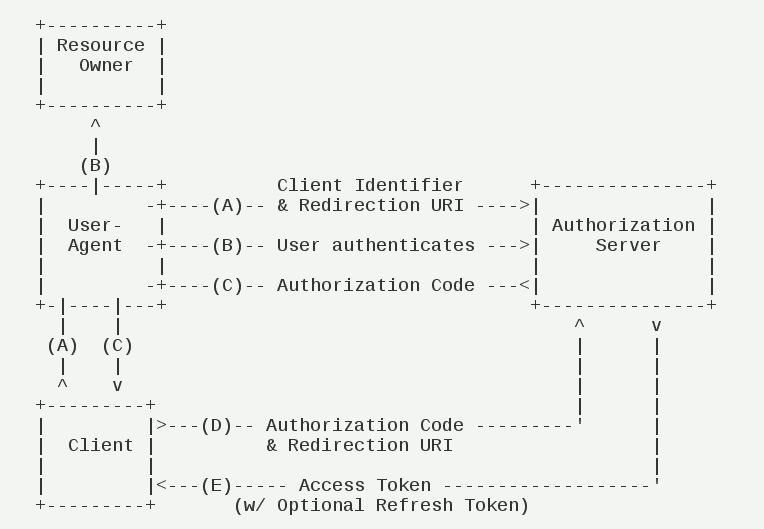
输入（即客户端请求，访问地址为Authorization Endpoint，以上步骤A）

* response\_type：表示授权类型，此处的值固定为"token"，必选项。
* client\_id：表示客户端的ID，必选项。
* redirect\_uri：表示重定向的URI，可选项。
* scope：表示权限范围，如需执行openid流程则必须包含openid。
* state：表示客户端的当前状态，可以指定任意值，认证服务器必须原样返回。

输出（即服务端响应，以上步骤C）

* access\_token：表示访问令牌，必选项。
* token\_type：表示访问令牌类型，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。
* 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。
* state：如果客户端的请求中包含这个参数，认证服务器的回应必须原样返回。
* id\_token：含有claim信息的JWT。必选项。

### 混合模式（Hybrid Flow）



（A）用户访问客户端，后者将前者导向认证服务器。

（B）用户选择是否给予客户端授权。

（C）假设用户给予授权，认证服务器将用户导向客户端事先指定的"重定向URI"（redirection URI），同时附上一个授权码。按照客户端请求的Response Type，也可同时返回id\_token、access\_token。

（D）客户端收到授权码，附上早先的"重定向URI"，向认证服务器申请令牌。这一步是在客户端的后台的服务器上完成的，对用户不可见。

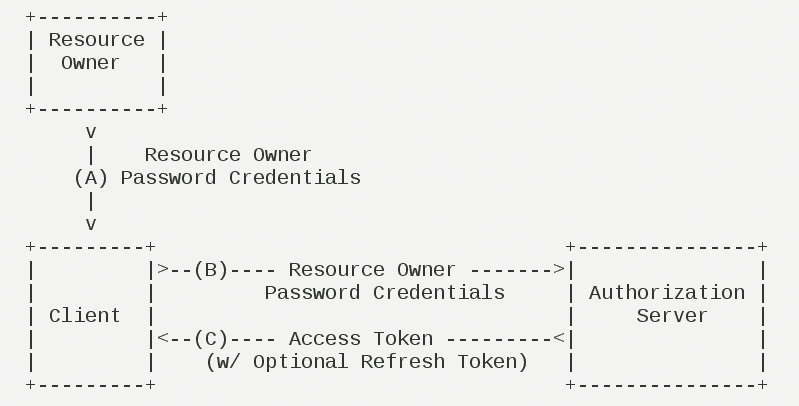
（E）认证服务器核对了授权码和重定向URI，确认无误后，向客户端发送访问令牌（access token）和更新令牌（refresh token）。

此流程与授权码模式很类似，区别在于C步骤返回的数据项。如果C步骤不给客户端提供授权码，流程和简化模式相同。如果所有令牌都从Token Endpoint返回，那么流程与授权码模式相同。

输入和输出上与授权码流程的区别：

* 输入参数中的response\_type：值可以是code id\_token、code token或code id\_token token
* 当输入参数的response\_type包含id\_token，响应值（输出参数）需包含id\_token；当输入参数的response\_type包含token，响应值（输出参数）需包含access\_token

### 密码模式（Resource Owner Password Credentials Flow）



（A）用户向客户端提供用户名和密码。

（B）客户端将用户名和密码发给认证服务器，向后者请求令牌。

（C）认证服务器确认无误后，向客户端提供访问令牌。

此流程中用户向客户端提供自己的用户名和密码。客户端使用这些信息，向"服务商提供商"索要授权。用户必须把自己的密码给客户端，但是客户端不得储存密码。

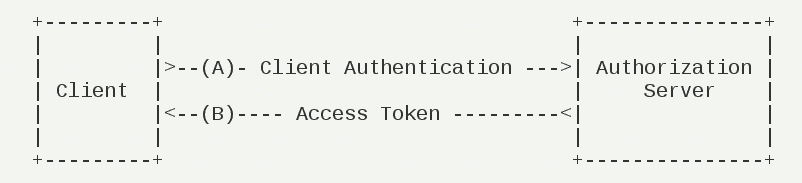
输入（即客户端请求，访问地址为Authorization Endpoint ，B步骤）

* grant\_type：表示授权类型，此处的值固定为"password"，必选项。
* username：表示用户名，必选项。
* password：表示用户的密码，必选项。
* scope：表示权限范围，可选项。

输出（即服务端响应，C步骤）

* 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（Bearer）。
* 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。
* client\_id：表示客户端ID，必选项。

### 客户端模式（Client Credentials Flow）



（A）客户端向认证服务器进行身份认证，并要求一个访问令牌。

（B）认证服务器确认无误后，向客户端提供访问令牌。

此流程指客户端以自己的名义，而不是以用户的名义，向"服务提供商"进行认证。严格地说，客户端模式并不属于OAuth框架所要解决的问题。在这种模式中，用户直接向客户端注册，客户端以自己的名义要求"服务提供商"提供服务，其实不存在授权问题。

输入（即客户端请求，访问地址为Authorization Endpoint，A步骤）

* grant\_type：表示授权类型，此处的值固定为"client\_credentials"，必选项。
* scope：表示权限范围，可选项。

输出（即服务端响应，B步骤）

* 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（Bearer）。
* 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。

### 更新令牌

如果用户访问的时候，客户端的"访问令牌"已经过期，则需要使用"更新令牌"申请一个新的访问令牌。

输入（即客户端请求，访问地址为Authorization Endpoint，A步骤）

* grant\_type：表示授权类型，此处的值固定为"client\_credentials"，必选项。
* refresh\_token：表示早前收到的更新令牌，必选项。
* scope：表示权限范围，可选项。

输出（即服务端响应，B步骤）

* 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（Bearer）。
* 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。
* client\_id：表示客户端ID，必选项。

## 认证后的流程

Client获得访问令牌后，就可以从UserInfo Endpoint查询用户信息。对于Openid的业务，这一操作就是最后的流程步骤；如果是OAuth的业务，还会有其他步骤。

输入（客户端请求）

* Access Token：必选项，建议在HTTP的头部通过Authorization字段发送

输出（服务端响应）

* 响应内容为JWT形式，其claims部分包含用户信息。用户信息的具体属性，以之前授权步骤中申请的scope为准。

## Database Schema

Users：用户表。

用户表可以不是本系统的一部分，系统可以通过其他方式获取验证信息。

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Type | Null | Key | Default | Comment |
| userid | varchar(20) | No | PK |  |  |
| username | varchar(20) | No |  |  | 需加盐及不可逆变换后存储 |
| password | varchar(40) | No |  |  |  |
| email | varchar(40) | No |  |  |  |
| email\_verified | tinyint(1) | No |  |  |  |
| valid | tinyint(1) | No |  |  |  |
| create\_date | timestamp | No |  | CURRENT\_TIMESTAMP |  |
| create\_uid | varchar(20) | No |  |  |  |
| update\_date | timestamp | Yes |  |  |  |
| update\_uid | varchar(20) | Yes |  |  |  |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Clients：客户端表  本文中的客户端指接入统一认证系统的应用   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | Field | Type | Null | Key | Default | Comment | | client\_id | bigint(20) | No | PK |  |  | | redirect\_uri | varchar(200) | Yes |  |  |  | | grant\_types | varchar(40) | No |  |  |  | | dyn\_reg | tinyint(1) | No |  |  | 是否动态注册，扩展用 | | valid | tinyint(1) | No |  |  |  | | create\_date | timestamp | No |  | CURRENT\_TIMESTAMP |  | | create\_uid | varchar(20) | No |  |  |  | | update\_date | timestamp | Yes |  |  |  | | update\_uid | varchar(20) | Yes |  |  |  | | client\_secret | varchar(20) | Yes |  |  | 验证客户端有效性用，此处仅记录密码。密钥另记录。 | | scope |  | No |  |  | 允许客户端访问的范围。 | |  |  |
| Authorization\_codes：授权码表  记录已经颁发的授权码   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | Field | Type | Null | Key | Default | Comment | | code | varchar(20) | No | PK |  |  | | client\_id | bigint(20) | No |  |  |  | | user\_id | varchar(20) | No |  |  |  | | redirect\_uri | varchar(200) | Yes |  |  |  | | expires | timestamp | No |  |  |  | | scope | varchar(200) | No |  |  |  | | create\_date | timestamp | No |  | CURRENT\_TIMESTAMP |  | |  |  |
|  |  |  |
| Access\_tokens：访问令牌表  记录已经颁发的访问令牌   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | Field | Type | Null | Key | Default | Comment | | access\_token | varchar(40) | No | PK |  |  | | client\_id | bigint(20) | No |  |  |  | | user\_id | varchar(20) | No |  |  |  | | expires | timestamp | No |  |  |  | | scope | varchar(200) | No |  |  |  | | token\_type | varchar(10) | No |  |  |  | | create\_date | timestamp | No |  | CURRENT\_TIMESTAMP |  | |  |  |
|  |  |  |
| Refresh\_tokens：刷新令牌表  记录已经颁发的刷新令牌   |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | Field | Type | Null | Key | Default | Comment | | refresh\_token | varchar(40) | No | PK |  |  | | client\_id | bigint(20) | No |  |  |  | | user\_id | varchar(20) | No |  |  |  | | expires | timestamp | No |  |  |  | | scope | varchar(200) | No |  |  |  | | create\_date | timestamp | No |  | CURRENT\_TIMESTAMP |  | |  |  |

Scopes：范围表

系统所有的范围的字典表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Type | Null | Key | Default | Comment |
| scope | varchar(20) | No | PK |  |  |
| is\_default | boolean | No |  | false |  |
| create\_date | timestamp | No |  | CURRENT\_TIMESTAMP |  |
| create\_uid | varchar(20) | No |  |  |  |
| update\_date | timestamp | Yes |  |  |  |
| update\_uid | varchar(20) | Yes |  |  |  |

## 现有系统情况

## 接口

现有的系统接口都是一次调用直接获得授权令牌。

**客户端授权（**类似Client Credentials Flow，获取的access\_token角色是client**）**

**输入参数**

|  |  |  |
| --- | --- | --- |
| **参数** | **说明** | **必填** |
| grant\_type | 固定值：client\_credentials | 是 |
| Authorization | Basic Base64("client\_id:client\_secret") | 是 |

此处的Authorization 不是HTTP Basic，而是application/x-www-form-urlencoded的形式传递。

**输出参数**

|  |  |
| --- | --- |
| **参数** | **说明** |
| access\_token | 有效期90天 |
| token\_type |  |
| expires\_in | 有效时间 单位秒 |

**password授权（**类似Implicit Flow，获取的access\_token角色是user**）**

**输入参数**

|  |  |  |
| --- | --- | --- |
| **参数** | **说明** | **必填** |
| grant\_type | password(固定值) | 是 |
| username | 用户名@企业后缀/手机号/电子邮箱 | 是 |
| password | 用户密码 | 是 |
| Authorization | Basic Base64("client\_id:client\_secret") | 是 |

**输出参数**

|  |  |
| --- | --- |
| **参数** | **说明** |
|  | 样例：{"access\_token":"{access\_token}","token\_type":"bearer","expires\_in":2591999} |

**客户端授权（**代理模式，获取的access\_token角色是client，代理访问其他认证中心**）**

**输入参数**

|  |  |  |
| --- | --- | --- |
| **参数** | **说明** | **必填** |
| grant\_type | client\_credentials | 是 |
| Authorization | Basic Base64("{client\_id}:{client\_secret}") | 是 |
| token\_url | 请求的链接地址 https://ucenter.shipxy.com/OAuth/Token | 是 |

**输出参数**

|  |  |
| --- | --- |
| **参数** | **说明** |
| access\_token | 有效期90天 |
| token\_type |  |
| expires\_in | 有效时间 单位秒 |

## 数据

**CompanyInfo企业信息表**

|  | **Name** | **Data Type** | **Max Length (Bytes)** | **Allow Nulls** | **Default** | **Description** |
| --- | --- | --- | --- | --- | --- | --- |
| pk | CompanyID | char(36) | 36 | no |  | 企业ID |
| 以下为企业信息，略 | | | | | | |

**CompanyUser企业用户信息表（作用类似Users表）**

|  | **Name** | **Data Type** | **Max Length (Bytes)** | **Allow Nulls** | **Default** | **Description** |
| --- | --- | --- | --- | --- | --- | --- |
| pk | UserID | char(36) | 36 | no |  | 用户ID |
| fk | CompanyID | char(36) | 36 | no |  | 企业ID |
| uk | UserName | varchar(100) | 100 | no |  | 登录名 |
|  | UserPwd | char(32) | 32 | no |  | 登录密码 |
|  | IsAdmin | tinyint | 1 | no | ((0)) | 是否管理员 |
|  | Enabled | tinyint | 1 | no | ((1)) | 启用/禁用 |
|  | Remark | nvarchar(50) | 100 | no | ('') | 备注 |
|  | CreateTime | datetime | 8 | no | (getdate()) | 创建时间 |
|  | Mobile | varchar(50) | 50 | no | ('') | 手机号 |
|  | Email | varchar(50) | 50 | no | ('') | 电子邮件 |
|  | openid | varchar(50) | 50 | no | ('') | openid（业务系统通过openid操作，不开放userid） |

**AuthInfo认证信息表（作用类似access\_token和authentication\_code表，code部分实际未用到）**

|  | **Name** | **Data Type** | **Max Length (Bytes)** | **Allow Nulls** | **Identity** | **Default** | **Description** |
| --- | --- | --- | --- | --- | --- | --- | --- |
| pk | Id | int | 4 | no | 1 - 1 |  | 编号 |
| fk | UserName | varchar(100) | 100 | no |  |  | 用户名 |
| fk | OpenId | varchar(100) | 100 | no |  |  | openid |
|  | ClientId | varchar(50) | 50 | Yes |  |  | 授权客户端ID |
|  | AuthorizationCode | varchar(100) | 100 | Yes |  |  | 授权Code |
|  | Ticket | varchar(500) | 500 | Yes |  |  | 票据号 |
|  | CodeExpTime | datetime | 8 | Yes |  |  | Code过期时间 |
|  | AccessToken | varchar(500) | 500 | Yes |  |  | 授权Token |
|  | TokenExpTime | datetime | 8 | Yes |  |  | 授权Token过期时间 |
|  | CreateTime | datetime | 8 | no |  | (getdate()) | 创建时间 |

**ElaneSystem系统配置表（作用类似Clients表）**

|  | **Name** | **Data Type** | **Max Length (Bytes)** | **Allow Nulls** | **Default** | **Description** |
| --- | --- | --- | --- | --- | --- | --- |
| pk | ElaneSysID | char(36) | 36 | no |  | Elane业务系统ID |
|  | ElaneSysName | nvarchar(50) | 100 | no |  | Elane业务系统名称 |
|  | ElaneSysKey | char(36) | 36 | no |  | Elane业务系统Key |
|  | SSOLogInURL | varchar(300) | 300 | no |  | 业务系统登录URL |
|  | SSOLogOutURL | varchar(300) | 300 | no |  | 业务系统注销URL |
|  | Enabled | tinyint | 1 | no | ((1)) | 启用/禁用 |
|  | BindCPU | tinyint | 1 | no | ((0)) | 是否绑定CPU |
|  | BindAdapter | tinyint | 1 | no | ((0)) | 是否绑定网卡 |
|  | CreateTime | datetime | 8 | no | (getdate()) | 创建时间 |
|  | AppId | varchar(50) | 50 | Yes |  | 分配给业务系统的appid |
|  | AppKey | varchar(50) | 50 | Yes |  | 分配给业务系统的appsecret |
|  | RedirectUrl | varchar(300) | 300 | Yes |  | 成功授权后的回调地址 |

## 关于用户的角色

## 角色相关概念

* 角色Role属于特定的企业；用户也属于特定的企业；
* 当用户有1~n个Role，默认具有普通用户的Role；
* 客户端不属于特定的企业；
* Role的功能范围解释权归客户端（应用）。

## 角色相关的表

CompanyInfo：企业信息表

此表可沿用原有企业信息表。

Roles：角色定义表

系统所有的角色的字典表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Type | Null | Key | Default | Comment |
| id | int | No | PK |  |  |
| role | varchar(20) | No |  |  |  |
| Company\_id | varchar(40) | No | FK |  |  |
| is\_default | boolean | No |  | false |  |
| create\_date | timestamp | No |  | CURRENT\_TIMESTAMP |  |
| create\_uid | varchar(20) | No |  |  |  |
| update\_date | timestamp | Yes |  |  |  |
| update\_uid | varchar(20) | Yes |  |  |  |

UserRoles：用户角色关系表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Type | Null | Key | Default | Comment |
| user\_id | varchar(20) | No | PK |  | 联合主键 |
| role\_id | varchar(20) | No | PK |  | 联合主键 |
| company\_id | varchar(40) | No | FK |  | 冗余，方便查询 |
| create\_date | timestamp | No |  | CURRENT\_TIMESTAMP |  |
| create\_uid | varchar(20) | No |  |  |  |
| update\_date | timestamp | Yes |  |  |  |
| update\_uid | varchar(20) | Yes |  |  |  |

## 代办操作相关的表

Delegations：代办授权信息表

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field | Type | Null | Key | Default | Comment |
| id | int | No | PK |  |  |
| user\_id | varchar(20) | No | FK |  |  |
| delegated\_userid | int | No | FK |  |  |
| delegated\_roleid | int | No |  | false |  |
| expires | timestamp | No |  |  |  |
| create\_date | timestamp | No |  | CURRENT\_TIMESTAMP |  |
| create\_uid | varchar(20) | No |  |  |  |
| update\_date | timestamp | Yes |  |  |  |
| update\_uid | varchar(20) | Yes |  |  |  |

## 其他事项

## 服务发现

如果需要支持客户端自动注册，须实现服务发现功能。

## 客户端的验证

认证服务器需要支持HTTP Basic、在请求体发送参数（client\_id和client\_secret，不能在URI中发送，不推荐）、JWT（claims必须有iss、sub、aud、jti、exp）或其他方式，用于验证客户端的真实性。

## 用户端的验证

图形验证码。

双因子验证。双因子认证（2FA）是指结合密码以及实物（信用卡、SMS手机、令牌或指纹等生物标志）两种条件对用户进行认证的方法。

## 用户登录的UI

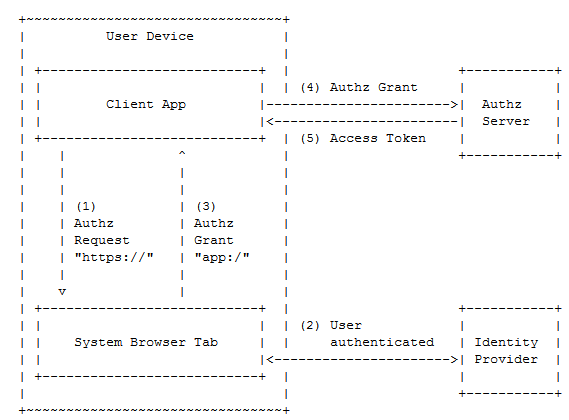
在验证码流程中，Client可以通过验证码请求的参数指定一下步的登录UI方式。参数名为display，取值范围包含page（页面跳转，默认）、popup（弹窗）、touch、wap。后两种一般用于移动设备。不推荐通过嵌入页面的方式展现登录UI（可利用X-Frame-Options来防止嵌入）。

## 用户登出

用户登出要区分在业务系统登出还是登出统一认证系统。

## 移动端

根据OAuth相关的建议，移动端的认证申请只能从应用外部用户代理（User agent）发起，例如系统浏览器（包括嵌入应用的浏览器）。（ OAuth 2.0 for Native Apps draft-wdenniss-oauth-native-apps-02）



1) 移动应用开启浏览器发出认证申请

2) 服务器认证用户, 有可能会关联其他认证系统。认证成功后返回认证码。

3) 通过定制的协议（例如app://），浏览器将焦点切换回移动应用，认证码通过协议URI的参数进行传递。

4) 移动应用通过认证码和证书访问服务申请令牌

5) 服务端发放访问令牌

可以看到以上流程是符合授权码模式的主要流程的。

## 用户信息来源与用户注册

严格讲，用户信息来源和注册不是认证系统的一部分。

## 附录1：用例与要求

1. 支持OpenID code flow
2. 支持全平台和单一租户，也就是亿海蓝其他支持OpenId的第三方软件可以用我们的ID 服务
3. 提供账户管理API（restful & json)
4. 提供账户管理 Command Line Interface
5. 支持API Token
6. 提供身份认证Web UI
7. 密码更改

用例与要求的说明：

* 按Openid和OAuth的规范实现即可满足以上1、2、5；
* 账户和密码管理的管理（含API、Cmd Line Interface）建议保留在现有企业用户中心中，身份认证Web UI建议放在统一认证中心实现；

## 附录2：安全上的考虑

1. https
2. Captcha （本机第一次，或失败后）
3. Password 加密 （double SHA256 Hash）
4. 非跳转身份认证安全机制 （参考 mySQL）
5. Auditability
6. 限制Client IP per user
7. 双因子认证 per Tenant / per service
8. 代办操作授权

安全相关说明

* https：统一认证系统的基本要求，必须实现；
* 密码加密存储：密码加盐后进行double SHA256 Hash处理。必须实现；
* 统一认证系统本身必须实现操作日志记录以便后续审计；
* Captcha和双因子认证作为安全加固选项，可延后实现；
* 用户登录IP限制：一般不应超过2个。现实中确有单用户同时使用多IP的情况。例如企业多个出口，或用户同时使用固定和移动网络。当出现用户多IP登录时，已登录的session如何处理、如何通知Client，需要有明确的策略。可能会涉及到session的管理（Openid有关于session的可选规范）。具体待定。
* 使用中要求一个人一个账户，不推荐使用多人共用账户的情况。如有某用户A替其他用户B进行业务操作的场景，需先有B或管理员对A授权代办，然后A用自己的账户完成功能。代办信息在统一授权系统中是一种scope。
* 非跳转身份认证安全机制。适用于非BS应用（？此处待确认）

1. 采用密码模式或公钥证书方式。
2. 采用类似移动应用的模式，在系统中注册一个特定的协议（如myapp://），然后在后续步骤使用这个协议指定redirect uri。可以结合TSL通讯防止code外泄。
3. 扩展的授权方式（OAuth 2.0协议 4.5）