**应急管理移动应用接入**

**技术规范**

**V2.0**

**二〇二四年四月**

**目 录**

[第1章 范围 7](#_Toc166180556)

[第2章 规范性引用文件 8](#_Toc166180557)

[第3章 术语和定义 9](#_Toc166180558)

[第4章 系统整体描述 10](#_Toc166180559)

[4.1 系统整体描述 10](#_Toc166180560)

[4.2 系统注册与登录界面 10](#_Toc166180561)

[4.3 用户身份认证方案 14](#_Toc166180563)

[第5章 前端H5改造 15](#_Toc166180564)

[5.1 应用接入流程 15](#_Toc166180565)

[5.2 H5应用开发 16](#_Toc166180570)

[5.3 统一身份认证对接 59](#_Toc166180576)

[5.4 VPN服务调用 61](#_Toc166180580)

[第6章 部政务用户统一身份认证 62](#_Toc166180581)

[6.1 统一身份认证场景 62](#_Toc166180582)

[6.2 应用系统对接流程 63](#_Toc166180586)

[6.3 接口描述 64](#_Toc166180587)

[6.4 认证SDK集成 69](#_Toc166180594)

[第7章 部政务用户统一身份认证身份信息同步接口 92](#_Toc166180599)

[7.1 应用系统集成 92](#_Toc166180600)

[7.2 接口描述 95](#_Toc166180604)

[第8章 部自然人、法人统一身份认证 102](#_Toc166180608)

[8.1 自然人、法人登录时序图 102](#_Toc166180609)

[8.2 新增自然人信息 103](#_Toc166180610)

[8.3 新增法人信息 104](#_Toc166180614)

[8.4 登录认证 106](#_Toc166180618)

[8.5 获取短信验证码 107](#_Toc166180622)

[8.6 Token验证 108](#_Toc166180626)

[8.7 获取用户信息 109](#_Toc166180630)

[第9章 附录 112](#_Toc166180634)

[9.1 H5改造-聊天消息结构附录 112](#_Toc166180635)

[9.2 部政务人员身份认证附录 119](#_Toc166180648)

[9.3 部政务人员身份认证信息同步附录 122](#_Toc166180652)

**文档说明**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **文档名称** | 《应急管理政务移动应用接入技术规范》 | | | |
| **内容描述** | 描述了应急管理政务移动应用整合接入技术要求。 | | | |
| **修订历史** | | | | |
| **日 期** | **版本** | **修订者** | **修订说明** | **审核人** |
| 2024年4月22日 | V1.0.0 |  |  |  |
| 2024年5月9日 | V2.0.0 |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

# 范围

移动应用平台APP做为应急管理部移动业务的统一入口。

本规范规定了应急管理部的移动应用发布到移动应用平台APP时需要进行的前端H5的开发要求和与部身份认证系统的对接要求。

对于前端的H5开发部分主要是介绍了第三方H5应用上架到移动应急平台的流程，移动应急平台提供统一用户登录、统一鉴权认证、统一网络接入等基础能力，H5 SDK的能力，服务端的的开放能力等。

对于对接部统一身份认证主要介绍了移动应用在移动端应用接入时，政务人员，自然人和法人用户与部统一身份认证的开放接口以及集成开发规范，适用于移动应用和部统一身份认证的对接。

# 规范性引用文件

下列文件对于本文件的应用是必不可少的。凡是注日期的引用文件，仅所注日期的版本适用于于本文件。凡是不注日期的引用文件，其最新版本（包括所有的修改单）适用于本文件。

1. ZWFWC0103-2018 国家政务服务平台政务服务移动端建设要求
2. 国务院办公厅关于印发“互联网+政务服务”技术体系建设指南的通知（国办函〔2016〕108号）
3. 进一步深化“互联网+政务服务”推进政务服务“一网、一门、一次”改革实施方案（国办发〔2018〕45号）
4. RFC5210 Internet X.509 Public Key Infrastructure Certificate Management Protocol (CMP)
5. RFC4211 Internet X.509 Public Key Infrastructure Certificate Request Message Format (CRMF)
6. ISO/IEC 8824/8825:2002 Abstract Syntax Notation One (ASN.1) and ASN.1 Encoding Rules
7. GM/T 0010-2012 SM2 密码算法加密签名消息语法规范
8. GM/T 0015 基于SM2密码算法的数字证书格式规范
9. GM/T 0024 SSL VPN协议

# 术语和定义

|  |  |  |
| --- | --- | --- |
| **术语** | **缩略语** | **说明** |
| 应急管理业务移动应用 | 移动应用 | 相关部门根据业务需要开发的适用于数据采集、流程办理、数据展示等业务类移动办理系统。  移动应用均要改造成H5方式对接。 |
| 应急管理业务  移动平台APP | 移动应用平台APP | 为各个应急管理政务移动应用提供统一用户登录、统一应用门户、统一鉴权认证、统一网络接入的承载体。 |
| 统一用户登录 |  | 移动应用平台APP负责对接统一身份认证系统，实现用户登录功能，并提供移动应用静默登录机制，为移动应用提供用户身份确认功能。移动应用平台App提供App+Web版本的统一门户，支持在应用内和应用外访问移动应用。 |
| ★政务用户 |  | 在移动应用中，身份信息有人维护的政务人员。  包括部，省，市，县，乡的应急管理部门的政务人员；有专员负责身份信息维护的各类信息员。 |
| ★自然人用户 |  | 在移动应用中，身份信息无人维护，需要到系统中自行注册的用户。 |
| 统一鉴权认证 |  | 移动应用上架到移动应用平台APP中的工作台后，可根据业务需要，按部门、区域、职能等业务属性，对应用可见性进行管理，并进行访问权限控制。  统一鉴权是移动应用级别的控制，用户进入移动应用后，细分权限控制由移动应用系统负责。 |
| 统一网络接入 |  | 移动平台负责对接VPN服务，移动应用申请上架时，应提供网络环境信息及审批信息。移动过平台根据移动应用系统的部署环境，按需建立VPN连接，提高安全性，简化移动应用系统部署工作。 |
| OneTalk |  | 移动应用H5 JS SDK的种类 |
| appKey |  | 应用编码，应用系统在统一身份管理系统注册产生，作为应用系统在统一身份管理系统中的身份标识 |
| appPwd |  | 应用凭证，应用系统调用接口时使用。 |

# 系统整体描述

## 系统整体描述

移动应急平台将作为应急管理部移动应用的唯一门户。

移动应用都需要上架到移动应急平台上进行发布。

## 系统注册与登录界面

移动应急平台对自然人用户，法人用户和政务人员用户区分了不同的用户入口。

在用户注册或登录时，要根据不同的身份选择不同的登录入口。



1. **登录效果图**

### 登录后界面

根据登录用户的身份进行不同，会呈现不同的页面。

#### 自然人用户登录界面

自然人用户登录后，可以看到自然人用户可以注册并使用的移动应用。



1. **自然人用户登录效果图**

#### 法人用户登录界面

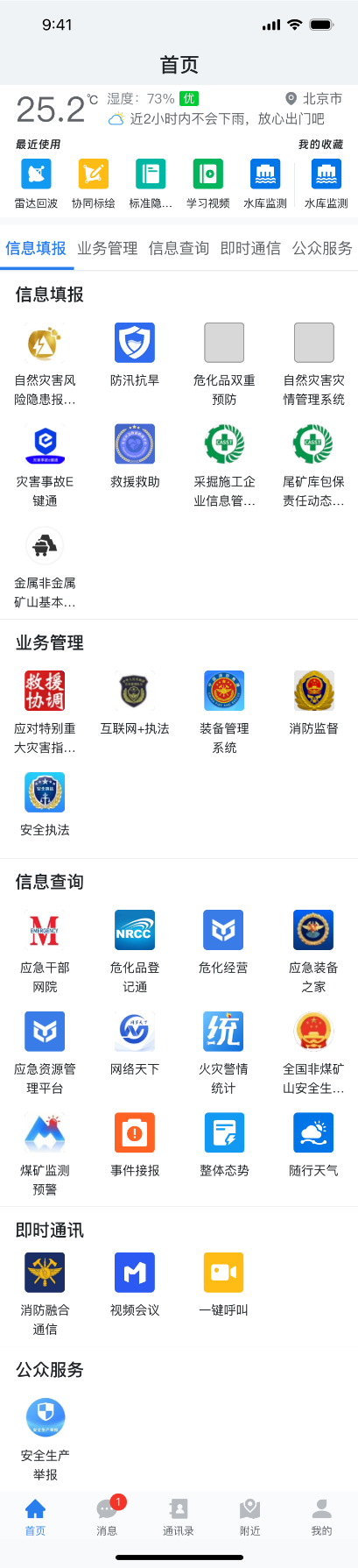
法人用户登录后，可以看到法人用户可以注册并使用的移动应用。



1. **法人用户登录效果图**

#### 政务用户登录界面

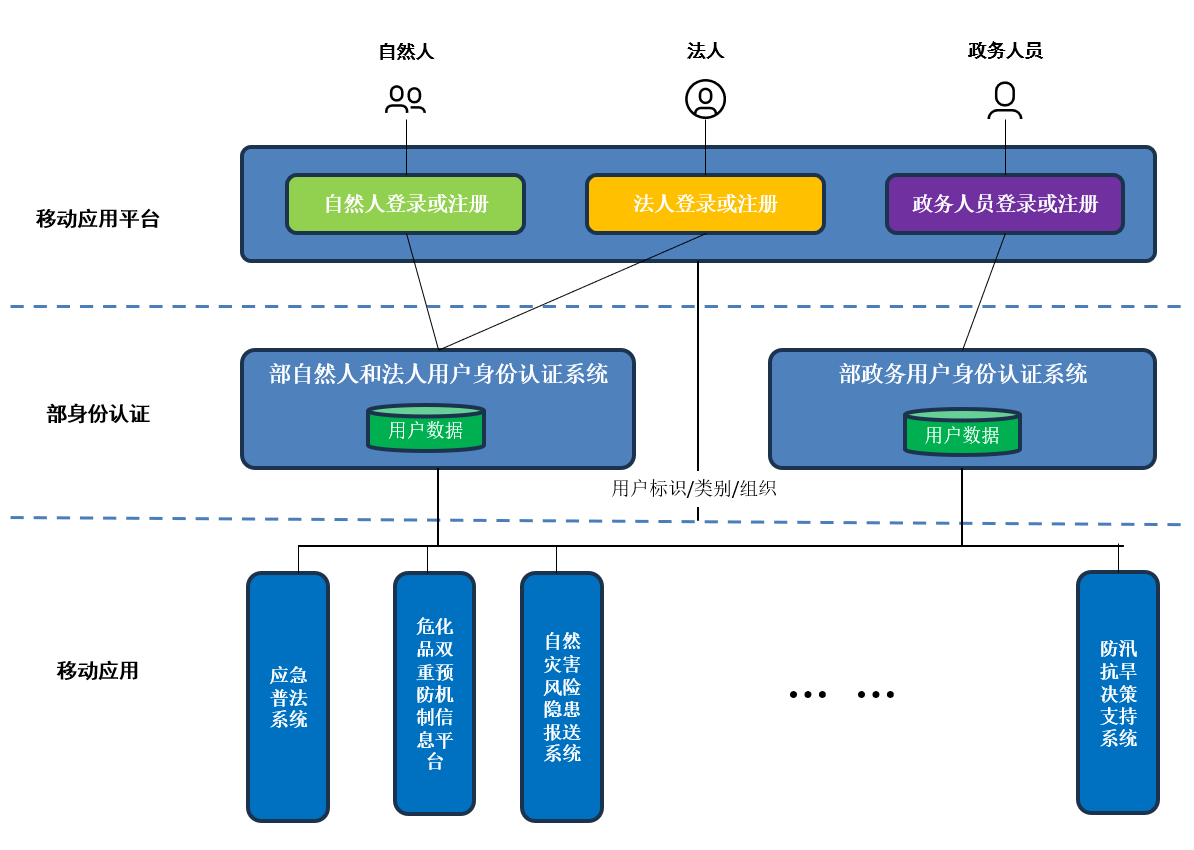
政务用户登录后，可以看到政务用户可以注册并使用的移动应用。



1. **政务人员用户登录效果图**

## 用户身份认证方案

对于用户身份的认证，对于政务人员的身份采用部政务人员统一身份认证，对于自然人和法人采用部自然人和法人统一身份认证。



1. **用户认证流程系统**

用户在选择不同的入口时，实际是选择的不同的认证系统。

# 前端H5改造

移动应急平台作为统一应用门户，支持第三方移动应用上架，为移动应用提供统一用户登录、统一鉴权认证、统一网络接入等基础能力。

## 应用接入流程

在移动应急平台上架移动应用，需要经过以下几个步骤：

### 申请开发者账号

（1）以邮件方式提交，开发商公司名称、管理员手机号码、开发者成员手机号。

（2）移动应急平台开通账号，邮件告知开发者账号ID。

（3）相关申请发到邮箱：onetalkdeveloper@linkcm.com。

### 新建移动应用

1. 邮件提交应用信息，创建应用所需资料如下：

**表 1资料表**

|  |  |
| --- | --- |
| **资料** | **注意事项** |
| 名称 | 必填，不得超过10个字 |
| 图标 | 必填，尺寸256x256 |
| 简介 | 必填 |
| 移动应用链接 | 必填，应用的地址 |
| 网络环境 | 必填，应用所在的网络，互联网、政务网。如果是政务网，需要提供审批信息。 |

（2）移动应急平台创建移动应用，邮件告知AppID、AppSecret等信息。

### 移动应用内测

（1）应用未正式上架之前，只有开发者内部成员可以看到移动应用。

（2）在移动应急APP-我的-开发者工具，可看到内测移动应用列表。

### 申请上架

（1）邮件提交要上架的应用：应用名称、AppID、版本说明。

（2）移动应急平台进行测试、审核，通过后上架到工作台。

（3）用户登录移动应急APP后，可以看到工作台面板，在工作台会展示所属单位已经上架的移动应用列表，供单位成员使用，比如：应急响应、台风路径等。展示效果如下示意图：

****

1. **展示效果图**

## H5应用开发

### 开发介绍

#### 客户端

移动应急APP作为移动端的主要入口，提供了H5 JS SDK，通过JS SDK可以快速对接用户认证、调用设备能力、拍照和录像、文件和相册访问等能力；JS SDK能力详情参考《5.2.4 H5 JS SDK》 章节。

移动应急APP支持安卓(5.0及以上)、鸿蒙(鸿蒙4及以下)系统。终端设备系统更新时，一般能兼容移动应急APP。在终端设备新系统不兼容旧系统接口的特殊情况下，移动应急APP需要对新系统做适配升级。

#### 服务端

应用开发方根据需要，可调用开放平台的服务接口，主要包括：用户认证、即时消息等；详情参考《5.2.5 服务端接口》章节。

(1)服务端接口使用HTTPS协议、JSON数据格式、UTF8编码。

(2)访问后端接口时，将应用密钥放在x-access-token头中。

(3)服务端接口仅允许在后端调用，不允许将应用密钥传递到前端使用。

### UI设计标准

#### 总体要求

此次规范基于尺寸750\*1624为标准，所有尺寸及标注大小均为1倍数值，单位为px，可以以此为参考进行适配。

#### 布局规范

设计规范中将APP页面布局分为：移动端首页、移动端工作台页面、移动端顶部加导航页面、移动端多导航多标签页面。

移动端从尽可能减少沟通与适配的成本考虑，基本以组件纵向平铺的方式进行展示。统一采用的设计画板尺寸为750\*1624；以“4”为一个单位，来调整元素间距及尺寸；采用平铺布局左右各32px的空间，卡片布局左右各32px空间。



1. **布局样例**

#### 图标规范

移动端系统中应用到的图标分为：通用图标、工作台图标。

a）通用图标

移动端直接使用到的功能性图标，其他图标可借鉴图标库或按照图标设计标准自行设计。

b）工作台图标

用于工作台应用界面，可以根据实际使用需求进行添加。

##### 颜色规范

色彩是以主题色为基准利用HSB颜色模式进行调节，产生一个新的亮度色，让亮度、饱和度和冷暖趋向改变跟主题色进行对比，光原色和原色间ACS值为1.70误差不超过0.1。

工作台图标可针对应业务的基调确定图标的性格颜色。

##### 工作台



##### 工作台-颜色



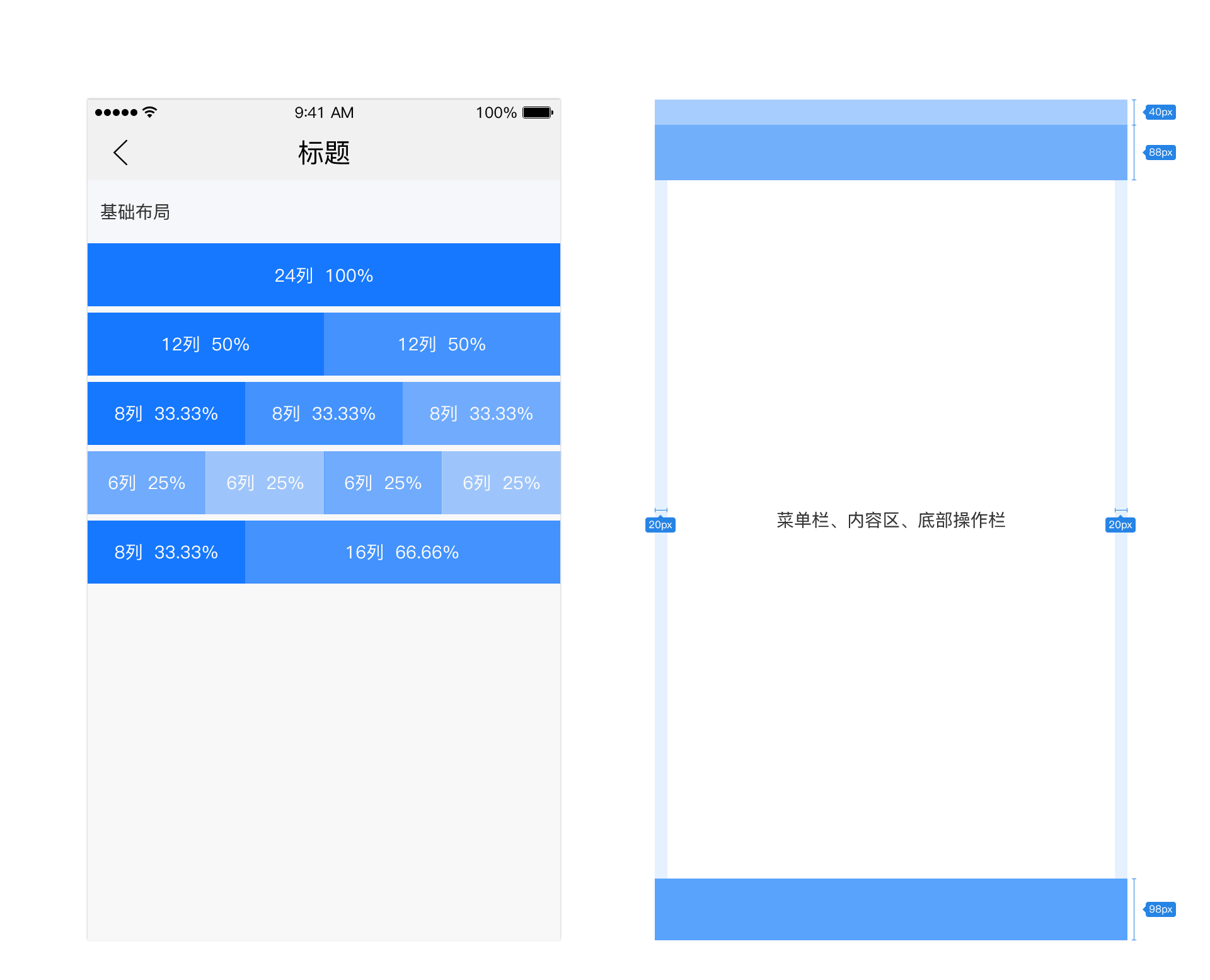
1. **工作台图标样例**

##### 工作台内容



1. **工作台内容**

##### 图标布局



1. **布局规范**

##### 尺寸规范

移动端系统工作台图标符号应符合以下最小尺寸设置要求：

a）符号大小：推荐80\*80像素、120\*120像素；

b）导出图片格式为jpg或png格式；

注：本文件对尺寸的规定基于可视分辨率，与比例尺寸无关。

##### 图标示例



1. **图样示例**

#### 应用规范

##### 布局规范

设计规范中将APP页面布局分为：移动端首页、移动端工作台页面、移动端顶部加导航页面、移动端多导航多标签页面。

移动端从尽可能减少沟通与适配的成本考虑，基本以组件纵向平铺的方式进行展示。统一采用的设计画板尺寸为750\*1624；以“4”为一个单位，来调整元素间距及尺寸；采用平铺布局左右各32px的空间，卡片布局左右各32px空间。



1. **布局样例**

##### 颜色规范

a）主题色

主题色Hex值为 #297EED，主题色是蓝色。

b）中性色

中性色适用于文本，背景和边框的颜色，通过运用不同的中性色，来表现层次结构。

c）功能辅助色

主题色之外的场景色，需要在不同的场景中使用（例如危险色表示危险的操作）。



1. **场景色样例**

##### 图表规范

a）柱状图

柱状统计图以时间或地市为维度进行数据展示，初始化时展示的最新日期或默认地市的数据，用户可以从右向左滑动查看其他数据。

b）折线图

基础折线图的一个类目只对应一个值，主要通过多系列数据配置的方式，展示同一类目下不同数据的变化，能够以折线和区域相结合的方式，智能地展示多维的数据变化趋势。

1. **图形样例**

##### 组件

a）按钮

按钮状态分为默认与点击2种状态，移动端使用场景的不同赋予大、小2种尺寸。

按钮类型分为主按钮、次按钮：主按钮突出“完成”、“确定”类操作；次按钮用于非主要动作；文字按钮弱化的按钮，采用更轻量的按钮样式；图标按钮应提供视觉线索，避免逐字阅读按钮文案，更高效地使用界面。

b）头像展示

移动端所有默认头像生成尺寸为56px，分为带圆角矩形头像、圆形头像2种；用户可自行上传照片更新头像。



1. **按钮与头像示例**

##### 弹窗规范

a）确认弹窗

弹出模态框，常用于消息提示、消息确认、在当前页面内完成特定的交互操作。

b）上拉菜单

弹出菜单，用于展示弹窗、信息提示、展示取消按钮等内容。



1. **弹窗与菜单**

#### 素材



1. **样例**

### 开发要求

#### 开发语言

前端使用 HTML、CSS 和 JavaScript 等前端技术构建 H5 页面的布局和样式。注意： 如原第三方应用是安卓原生App，需改造成H5，可根据自身业务指定改造计划，可将核心业务先改造完成上架到移动应用平台，逐步完成改造。

#### 应用可用性

##### 应用无缺陷

应用不存在等级致命和严重的缺陷，避免出现空白页、页面错乱等。

##### 应用易用性

输入异常提示、结果提示、输入方式、结果展示等应是清晰易理解的：

* 移动应用名称通俗易懂，应用类型与名称准确且匹配，如查询类应用叫XX查询、办理类应用叫 XX办理等；
* 如有需用户输入的信息，需校验输入的合法性，例如非空校验、格式校验等；
* 应用中包含有位置信息的业务场景，应根据实际情况使用地图定位、距离排序及导航功能；
* 若应用需根据用户属性进行使用，要求能够自行判断用户属性即可使用；
* 按照用户真实使用场景及需求，整合同类型、相关联的应用，若过于分散，不利于用户查找与使用。不允许同一类（项）应用的子项拆分为独立的应用；

不限于以上5条的其他易用性要求。

#### 应用安全性

##### 数据安全

移动应用需符合国家相关网络安全要求规范，做好网络安全保护工作；应用数据安全技术要求包括但不限于：

* 所有接入的移动应用经过安全加密、数据脱敏等处理，并支持 HTTPS 安全协议；
* 不应在数据存储、处理、传输过程中非法截留或窃取数据；
* 涉及敏感数据需加密存储，且要有合适的加密强度，不应使用弱加密，不应采用明文形式存储或网络传输敏感数据；
* 对个人信息、企业信息等关键信息的展示，强制要求脱敏处理；
* 应用如有数据删除功能，在删除数据前应明确提示用户，并由用户确认删除操作；
* 应采用安全机制保护账号、密码和支付信息等敏感数据的输入与显示，除非用户主动允许外，不应明文显示账号、密码等敏感数据，涉及安全要求较高的重要敏感信息，宜采用安全键盘作为数据的输入接口；
* 未经用户授权时，不应传输敏感数据，在得到用户授权后传输的敏感数据，应采用加密等安全技术进行传输，账号、密码传输要防止重放攻击和中间人攻击；
* 使用用户信息应遵循合法、正当、必要的原则，明示收集使用信息的目的、方式和范围，并经用户同意。

##### 代码安全

应用代码安全技术要求包括但不限于：

* 宜采取反编译防护、反盗版防护等措施防止被破解、篡改或二次打包，保护源代码安全；
* 所引用或包含接入方代码或开源代码，应确保接入方代码或开源代码安全性，应对已公布的安全漏洞及时更新；
* 代码中不应存在已公布的高危风险漏洞；
* 应禁止内部组件被外部程序调用，如需供外部调用，应检查调用者是否符合访问控制机制。

#### 数据准确性

服务提供的相关结果数据准确，应与实际业务场景匹配，并保持与接入方数据一致。

#### 应用系统稳定性

* 所有上架的移动应用接口响应快，能抗压抗并发，并准备故障解决预案。
* 每对移动应用进行1000次访问调用，服务故障率低于0.1%，响应异常次数不超过1次。

#### 应用兼容性

* 兼容性高，能够自动适配不同（主流）屏幕尺寸的手机机型；
* 交互提示友好，不同使用场景下要有对应的使用引导；
* 结构布局清晰，相同类型业务数据要归类展示，不同业务数据展示方式要有区分；
* 体验流畅高效，页面数据展示延时最长不超过 3s，页面切换平滑流畅；
* 操作层级简化，精简页面流程设计。

### H5 JS SDK

【使用说明】

* 第一步、引入H5 JS SDK脚本

1. **引入H5 JS SDK 脚本**

注意事项：jssdkBaseUrl待定

* 第二步、接口初始化

页面引入H5 JS SDK的脚本（ecpot.js）后，其实不会执行加载和初始逻辑，而是需要您手动启动初始逻辑。框架的初始接口是$c.init。执行初始接口后，在脚本准备完毕后，触发成功回调。

1. **接口初始化**

注意事项：

所有异步接口都基于jQuery的延迟对象Deferred实现异步回调，请按照该规范调用接口和获取数据。

* Deferred回调结果说明:

Done：成功返回（result不为空）。存在取消操作的接口，例如选择/获取联系人，若中 途失败，则返回空对象，表示失败

Fail：取消操作或发⽣异常。一般不需要处理这种情况。取消只发⽣在存在取消操作的接 口，其它都是异常

#### 认证鉴权

##### ecpot.getTicket(options)：获取票据

选项：

 options.appId 移动应用标识[必填]

 options.redirectUrl 移动应用地址，用于统—登录页完成授权后的重定向，默认会使用调用此接 口时的 location.href 值[选填]

 options.forceRedirect 是否强制重定向到统一登录页面（忽略URL中的a2\_ticket） [选填]

\*\*返回值： \*\*Promise<{ ticket: string }>

接口例子：

调用此接口时，如果URL中有a2\_ticket查询参数则会直接返回ticket，否则将跳转至统—登录 页，用户在统—登录页完成登录后重定向回移动应用页面（携带a2\_ticket查询参数）。

##### ecpot.logout(options)：登出账号

选项：

 options.appId 移动应用标识[必填]

 options.redirectUrl 移动用地址，用于统一登录页完成授权后的重定向，默认会使用调用此接口时的 location.href 值[选填]

接口例子：

1. **调用此方法会重定向至统一登录页。**

#### 窗口控制

##### ecpot.open(url,options,force)：打开新窗口

**选项：**

 url：网页链接接口支持相对和绝对路径的页面链接，具体参考标准的打开方式。

 options：窗口标题或扩展选项。若是对象，窗口标题可以配置在head属性（字符串类型）。

 页面刷新：框架覆盖了window.open的默认行为，请不要使用该接口刷新页面。页面刷新请使用 location.href。通过该接口，页面可以打开新的窗口（不是刷新当前页面）。

**接口例子：**

##### ecpot.close()：关闭窗口

**接口例子：**

##### 标题栏

###### ecpot.showTitleBar()：显示标题栏

**接口例子：**

###### ecpot.hideTitleBar()：隐藏标题栏

**接口例子：**

###### ecpot.setTitleRight(options)：设置标题栏右边按钮

选项：

 若参数options不为空，则配置右边按钮，否则移除右边按钮。

 options.button：按钮文字或图片；

 支持网络图片、内置图片： 'res://navi\_more'。

 options.onClick：点击事件回调。

 按钮文字建议使用2个汉字（或4个字母）。图片大小96x96，中间内容区域大小建议36x36， 其它区域建议保持透明。

**接口例子：**

###### ecpot.setTitleLeft(options)：设置标题栏左边按钮

**选项：**

 若参数options不为空，则配置左边按钮，否则移除左边按钮。

 options.button：按钮图片。

 按钮图片样式跟右边按钮一样。options.button还支持以下特殊图片： “ <”默认返回按钮， “x” 默认关闭按钮。

**接口例子：**

###### ecpot.lockTitleBack(strict, withLock)：锁定标题栏返回按钮

注意：点击返回按钮会触发返回键逻辑，而不是直接关闭

**选项：**

 目前只有安卓支持返回键和返回键拦截功能。返回键功能参考“页面容器 ”接口。

 若配置“window.\_ot\_lock\_title\_back=true”，则在容器初始后直接启动锁定。

 标题栏锁定后，不能刷新页面，否则会丢失标题栏处理逻辑，继而导致关闭按钮点击⽆效。若 标题栏已经锁定且需要刷新页面，应该在刷新前解锁标题栏或者最新页面也是锁定标题栏。

**严格模式：**

 若传递strict参数（可选较大数值，例如1000），则关闭严格历史检查机制（默认是开启严格 模式的）。

 ⾮严格历史机制下，锚点刷新也会记录进返回历史。这时，可以不用拦截返回键事件处理返回 逻辑。

 若配置“window.\_ot\_strict\_back\_count=数值”，则在容器初始后直接关闭严格历史机制。

 若传递withLock=true参数，则同时锁定标题栏。

**解锁接口：**

 ecpot.unlockTitleBack()：解锁标题栏返回按钮

**判断接口：**

 ecpot.isLockTitleBack()：是否锁定标题栏返回按钮

**接口例子：**

##### ⽔印

###### ecpot.showWatermark()：显示⽔印

**接口例子：**

###### ecpot.hideWatermark()：隐藏⽔印

**接口例子：**

##### ecpot.rotateWindow(options)：旋转窗口

**选项：**

 options.titleBar（可选）：显示标题栏，默认保持不变

 options.statusBar（可选）：显示状态栏，默认保持不变

 窗口默认方向是竖屏，即宽度小于⾼度。当前窗口方向可以通过窗口大小判断：

 window.innerWidth < window.innerHeight，即竖屏

 window.innerWidth > window.innerHeight，即横屏

 窗口方向改变过程中，可能存在多次大小调整，实际取决系统的刷新。

**接口例子：**

##### 页面容器事件处理

**注：在页面内可使用**$c**全局对象，访问页面容器**

$c.open(options)：拦截返回事件

$c.close()：手动关闭拦截

$c.getSublayerSize()：拦截事件数量

$c.getSublayerList()：拦截事件列表。注意，不要直接清空列表！

参数说明：

options.func：按键事件

options.autoClose：自动关闭，即只拦截一次！否则会拦截到用户手动关闭（默认） 注意，拦截行为是可以多次调用的，由调用栈管理先后顺序！

$c.open即⼊栈， $c.close即出栈

接口例子：

实例： back\_event

 实例： title\_back

#### 位置/地图

##### ecpot.getLocation(options)：获取地理位置

选项：

 options.pick（可选）： true表示选择位置，否则返回当前位置

 options.force（可选）： true表示强制刷新位置，否则1分钟内只返回上次定位结果，超过1分 钟才会重新获取位置（默认）

 options.timeout（可选）：定位的等待超时时长，单位秒，默认30秒

 options.expires（可选）：定位的缓存有效时长，单位秒，默认60秒

 options.needAddress（可选）： true表示逆地理编码，否则只返回经纬度。默认都会进行逆 地理编码，如不需要请配置为false。

 若使用选择位置功能，则可以配置初始位置（默认定位到当前位置）。

 初始位置参数包括： latitude（纬度，必选）、longitude（经度，必选）和title（位置描述，可选）。

**坐标系：**

 WGS-84

**成功返回：**

 result.latitude：纬度

 result.longitude：经度

 result.title：地理位置

 result.street：街道名称

 result.streetNum：门牌号

 result.ad：地区名称

 result.adCode：地区编码

 result.city：城市名称

 result.cityCode：城市编码

 result.province：省份名称

**接口例子：**

##### ecpot.watchLocation(options)：周期性获取地理位置

**选项：**

 options.interval（可选）：循环定位的间隔，单位秒，默认60秒

 options.callback（可选）：循环定位的回调函数，参数包括result（latitude是纬度， longitude是经度）

 options.clear（可选）：关闭监听任务，其值是创建时返回的任务id。页面关闭后，其创建的任务会自动关闭

坐标系：

 WGS-84

**成功返回：**

 result：监听任务id，可用于关闭任务

**接口例子：**

##### ecpot.openNavigation(options)：打开导航地图

**选项：**

 options.to：终点经纬度，包括lat（纬度）和lng（经度）

 options.from（可选）：起点经纬度，包括lat（纬度）和lng（经度）

 options.plan（可选）：是否打开路径规划，否则自动规划并进⼊导航。默认为打开路径规划

 options.type（可选）：规划/导航类别（0：驾车， 1：公交， 2：步行， 3：骑行， 4：火车， 5：长途客车）

**地图应用：**

 应用会尝试打开手机安装的地图软件。若手机地图应用都不存在，则会打开网页导航。

**接口例子：**

##### ecpot.showMap(options)：显示地图，并可在指定位置增加标记信息

选项：

 options.latitude（可选）：标记的纬度

 options.longitude（可选）：标记的经度

 options.zoom （可选）：缩放级别（3~19），默认19

 options.title（可选）：标记的标题

 options.snippet（可选）：标记的描述

 options.barTitle（可选）：标题栏的标题

坐标系：

 WGS-84

接口例子：

#### 媒体/文件

##### ecpot.chooseMedia(options)：获取图片原始数据

从相册获取图片或拍照

选项：

 options.take（可选）：true表示拍照上传，否则选择相册图片

 options.multi（可选）： true表示相册多选，否则相册单选

 options.video（可选）： true表示包含视频（默认），否则只有图片

 options.front（可选）： true表示使用前置摄像头拍照，否则使用后置摄像头

 options.preview（可选）：不为空表示使用小窗预览，否则使用全屏预览

 options.control（可选）：控制小窗预览行为， -1：关闭，0：启动， 1：拍照， 2：上传， 3： 暂停， 4：切换

 相册多选能够压缩图片和编辑图片，而单选只能选择单张原始图片。使用拍照时，长按可以录像。

 小窗预览可自定义预览区域和选择拍照尺寸。更多介绍可打开“ 自定义拍照 ”页面，参考例子开发。

小窗预览参数：

 preview.top（上边偏移）、preview.left（左边偏移）

 preview.width（窗口宽度）、preview.height（窗口⾼度）

 preview.scale（预览缩放方式， 0：适配窗口（默认）， 1：同比例铺满窗口， 2：拉伸铺满窗口）

 preview.front（使用前置摄像头， true表示前置， false表示后置，默认前置）

 preview.camera.width（相机宽度，单位像素）、preview.camera.height（相机⾼度，单位像 素）

 preview.finder.offset（镂空大小偏移）、preview.finder.square（镂空形状， true表示正方形， false表示长方形）

**相机宽⾼：**

 若没有配置相机宽度和⾼度，则选择当前手机屏幕宽度最佳比率，例如7201280、 10801920。

 相机大小选择采用就近原则，即选择相同大小或跟指定大小比例最接近的尺寸。

 常用相机尺寸：

144176,240320,288352,360640,480720,480640,600800,6401280,7201280,9601280,1080 \*1920。

**镂空窗口：**

 若配置finder为空，则不显示镂空窗口。

 若配置finder为空对象，则启用默认的镂空窗口，即offset=0、square=true。

 默认finder的大小，正方形宽度约占屏幕0.625，长方形宽度约占屏幕0.75。

控制回调：

 在使用小窗预览时，接口执行后，会⽴即回调，其中result表示打开结果（true表示成功）。

 再次调用接口并传递“options.control”参数即可以完成预览的行为控制。

 控制接口的回调有两个参数，第一个参数result表示执行结果，第⼆个参数mode表示执行的控 制行为。

**成功返回：**

 result.data（base64编码）

 result.mime

 若是多选模式，则返回的是数组。数组元素内容跟单选内容一样

**失败返回：**

 result为空对象

 若是多选模式，则返回空数组

**接口例子：**

##### ecpot.chooseFile(options)：获取文件原始数据

**成功返回：**

 result.data（base64编码）

 result.mime

**失败返回：**

 result为空对象

**接口例子：**

##### ecpot.playVideo(options)：播放视频链接

选项：

 options.url：视频链接

 options.landscape（可选）：横屏模式

 options.type（可选）：播放引擎， 0系统引擎（默认） ,1增强引擎（支持hls和http-flv的H.265 编码）

 建议少用⾮跨平台的视频格式和浏览器专用协议。

**接口例子：**

##### ecpot.showDoc(url)：查看文档

**选项：**

 url：文档链接

 支持pdf、ppt 、pptx 、docx 、xls 、xlsx 、txt格式

**接口例子：**

#### 设备功能

##### ecpot.keepScreen(on)：开启或关闭屏幕常亮

**选项：**

 on ：是否开启屏幕常亮，默认false

**接口例子：**

##### ecpot.scanQRCode(options)：识别二维码图片

**选项：**

 options.open：是否自动打开链接内容，默认true

**成功返回：**

 result.text：识别文字

 result.open：是否已打开链接内容

**接口例子：**

##### ecpot.callNumber(options)：拨打电话

**选项：**

 options.number：拨打号码

**接口例子：**

##### ecpot.captureScreen(options)：截屏并标绘

**选项：**

 options.delay：截屏开始延迟，单位毫秒，默认3000。请留⾜够时间确认权限和切换页面 · options.edit：截屏后是否标绘，默认false

 options.reuse ：是否重新标绘最后截图，默认false

**成功返回：**

 result.data：图片数据，格式Int8Array

 result.mime：图片类别，原始截图是image/png，标绘结果是image/jpeg

 图片数据格式是Int8Array，可以直接进行字节处理和发送。需要显示时，可转成Blob： new Blob([result.data], {type: result.mime})。

 由于图片很大，不建议使用Data URL显示，请使用Blob URL： URL.createObjectURL(blob)。

**接口例子：**

##### ecpot.sampleSensor(options)：采集传感器数据

**选项：**

options.sensors ：传感器类型数组（[[sensor,error1,...],...]），支持的类型名称请参考例子， 例如ACCELEROMETER 、PROXIMITY 、LIGHT。没有该参数表示结束

 options.error ：计算方向的变化误差。若启动重力加速度（ACCELEROMETER），可以配置该 参数自动计算角度和方向（角度变化在误差内才触发方向变化）

 ptions.onSensor：传感器数据回调，参数包括sensor （类型名称）、value（数值数组）。数 值的作用可参考官方文档

 options.onAccelerometerChange：加速度数据回调，参数包括value（对象， x是x轴重力值， y是y轴重力值， z是z轴重力值。最大加速度0.98）

 ptions.onCompassChange：罗盘数据回调，参数包括value（对象， direction是角度。角度0 指向北， 90指向东， 180指向南，依此类推）

 options.onDeviceMotionChange：设备方向数据回调，参数包括value（对象， azimuth是绕着 z轴转动的角度，范围0到360， pitch是绕着x轴转动的角度，范围-180到180， roll是绕着y轴 转动的角度，范围-90到90）

 options.onGyroscopeChange：陀螺仪数据回调，参数包括value（对象， x是绕着x轴转动的 角速度， y是绕着y轴转动的角速度， z是绕着z轴转动的角速度）

规范类型：

 onAccelerometerChange、onCompassChange、onDeviceMotionChange、

 onGyroscopeChange是全平台规范类型，只需配置回调即可启用，不用再配置sensors选项。

**未规范类型：**

 其它传感器类型并未全平台规范。

**成功返回：**

 result：跟传递的类型数组索引一一对应的Boolean数组，表示指定传感器是否注册成功

 传感器的数值作用可以参考以下官方文档：

<https://developer.android.google.cn/reference/android/hardware/SensorEvent?hl=en>

 若启动重力加速度（ACCELEROMETER）且配置自动计算角度，则数值数组前面两个分别是 方向和角度，后面是原始的重力数值。

**接口例子：**

#### AI增强

##### ecpot.dictateVoice(options)：语⾳听写

**选项：**

 options.onBegin（可选）：开始回调（按住按钮事件）

 options.onResult（可选）：识别回调（识别结束事件）。回调只返回最新识别的结果

 options.onEnd（可选）：关闭回调

 options.color（可选）：关闭按钮颜⾊（默认灰⾊#222）

 配置回调函数的情况下，控件只保留点击按钮和关闭按钮，其它区域都不显示。

**成功返回：**

 result.text：识别文字

**接口例子：**

#### 分享

##### ecpot.share(options)：分享内容

**选项：**

 options.type（可选）：分享渠道。 0：微信好友（默认）； 1：微信朋友圈； 2：剪切板； 3： 系统应用； 4：移动应急APP

 options.text（可选）：分享标题。若分享到系统应用时，这个字段会跟内容、链接合并成一 段文本

 options.desc（可选）：分享内容。若分享到系统应用时，这个字段会跟标题、链接合并成一 段文本

 options.url（可选）：分享链接。若拷⻉到剪切板时，只使用这个字段

 options.thumb（可选）：分享缩略图链接。目前只有使用微信渠道且包含分享链接时有效

**接口例子：**

#### 示例

【引用JS SDK脚本和调用接口的示例】

### 服务端接口

服务端开放接口工作在HTTP协议之上，消息格式为application /json，采用utf-8编码。

**基本约定：**

 接口采用RESTful⻛格，一般情况下，请求及响应消息体均为json格式。

 接口双方遵循HTTP内容协商机制，请求方应负责在Accept和Content-Type头中设置正确的 值，本文中的服务端接口，如⽆特别说明，Accept头都应设置为： application/json

 请求参数通过body传递时，要求使用json格式

 除⾮有单独的约定，否则所有接口都返回2xx的HTTP状态码，并通过x-result-code, x-result-message头字段返回接口调用的错误码和描述信息；一般情况下，x-result-code基本约定如下：

 x-result-code=0，接口调用成功，没有x-result-code头时，视为x-result-code=0

 x-result-code=-1，传⼊参数不符合要求

 x-result-code>0，业务层的错误码，参考接口定义

**表 2**

|  |  |  |
| --- | --- | --- |
| 名称 | 数据类型 | 说明 |
| x-result-code | int | 操作返回结果，为空或0：成功，其他表示失败 |
| x-result-message | string | 操作失败，返回相应的提示信息 |

**基本数据类型：**

 string：字符串

 short： 16位整型

 int： 32位整型

 long： 64位长整型

 float： 32位单精度浮点型

 double： 64位双精度浮点型

 boolean：布尔型

 byte：字节

 bytes：字节数组

 map： key-value数据结构

**接口鉴权机制**

 请求必须携带有⾜够权限的访问令牌才能访问接口。访问令牌存放在x-access-token头中

 x-access-token参数填写分配的AppSecret

 服务端鉴权失败时，将HTTP状态码设为412，并通过x-a2-code头返回鉴权层的错误码。

 x-a2-code=1：令牌已过期，应用应启动【刷新访问令牌】流程

 x-a2-code=2：令牌不存在，应用应启动【认证（登录）】流程

 -a2-code=3：令牌权限不⾜

**联调服务器**

 Base URL定义

 待定

#### 认证鉴权

##### 使用票据获取用户信息

 调用方法： PUT {Base URL}/open/a2/ticket

 权限要求： apibase

###### 请求

**请求消息示例**

###### 响应

**x-result-code定义**

 0：创建成功

 其它：失败

**响应消息**

 x-result-code=0时，返回map类型，包含以下字段

 openUserId： string类型，和appId相关的用户唯—标识

 userName： string类型，用户名

 mobileMd5s： string数组，用户手机号码列表，仅提供Md5摘要，调用方可用于和自 有用户匹配关联

响应消息示例

#### 消息服务

##### 创建模板

 调用方法：POST {Base URL}/open/mq/template

 权限要求：apibase

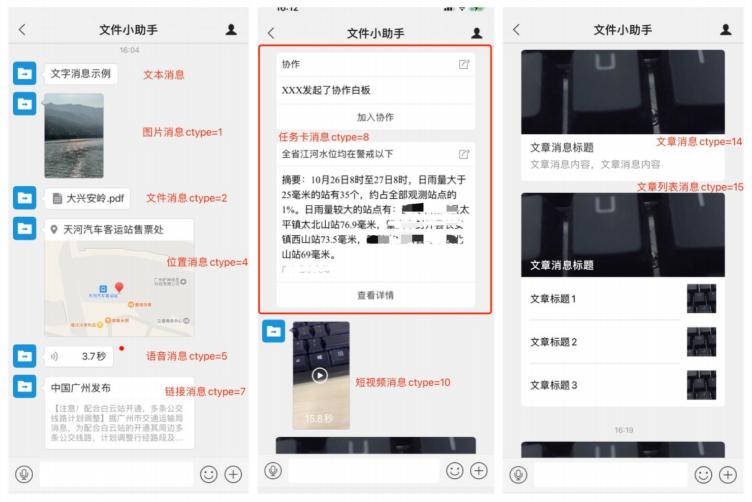
###### 请求

**表 3**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 参数名称 | 是否必须 | 参数类型 | 参数位置 | 说明 |
| title | yes | string | body | 模板名称 |
| channel | no | string | body | 消息下发渠道 |
| contentType | no | short | body | 模板内容类型， 0：普通文本； 1：  Freemarker模板 |
| content | no | any | body | 模板内容 |

Freemarker模板内容示例，更多参考《6.1 聊天消息结构章节》

**消息效果图示例：**



###### 响应

x-result-code定义：

 0：成功

 其它：失败

响应消息

 x-result-code=0时，返回map类型，包含以下字段

 templateId：模板标识，string类型

##### 修改模板

 调用方法：PUT {Base URL}/open/mq/template/{templateId}

 权限要求：apibase

###### 请求

**表 4**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 参数名称 | 是否必须 | 参数类型 | 参数位置 | 说明 |
| templateId | yes | string | path | 模板标识 |
| title | no | string | body | 模板名称 |
| channel | no | string | body | 消息下发渠道 |
| contentType | no | short | body | 模板内容类型， 0：普通文本； 1：  Freemarker模板 |
| content | no | any | body | 模板内容 |

###### 响应

**x-result-code定义**

 0：成功

 1：没有权限

 2：模板不存在

**响应消息**

 没有消息内容

##### 删除模板

调用方法： DELETE {Base URL}/open/mq/template/{templateId}

 权限要求： apibase

###### 请求

**表 5**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 参数名称 | 是否必须 | 参数类型 | 参数位置 | 说明 |
| templateId | yes | string | path | 模板标识 |

###### 响应

**x-result-code定义**

 0：成功

 1：没有权限

 2：模板不存在

**响应消息**

 没有消息内容

##### 获取模板信息

 调用方法： GET {Base URL}/open/mq/template/{templateId}

 权限要求： apibase

###### 请求

**表 6**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 参数名称 | 是否必须 | 参数类型 | 参数位置 | 说明 |
| templateId | yes | string | path | 模板标识 |

###### 响应

**x-result-code定义**

 0：成功

 1：没有权限

**响应消息**

 x-result-code=0时，返回Template结构

##### 查询模板列表

 调用方法： GET {Base URL}/open/mq/templates?keyword={keyword}&from={from}&count= {count}

 权限要求： apibase

###### 请求

**表 7**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 参数名称 | 是否必须 | 参数类型 | 参数位置 | 说明 |
| keyword | no | string | query | 搜索关键词 |
| from | no | int | query | 期望返回的起始行 |
| count | no | int | query | 期望返回的行数，默认为10 |

###### 响应

**x-result-code定义**

 0：成功

 其它：失败

**响应消息**

 x-result-code=0时，返回map结构，包含以下信息：

 elements： Template列表

 total：符合条件的总行数

 from：当前返回的起始行

##### 发送消息

 调用方法： POST {Base URL}/open/mq/message?channel={channel}  权限要求： apibase

###### 请求

**表 8**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 参数名称 | 是否必须 | 参数类型 | 参数位置 | 说明 |
| channel | yes | string | query | 发送渠道 |
| receivers | no | string[] | body | 消息接收方 |
| data | yes | map | body | 消息数据 |

channel=chat，IM聊天消息，聊天消息支持模板方式，应用可以先创建好消息模板，之后发送消息时，只需要传⼊ templateId和相关的params信息，可以提⾼消息的发送效率

常用的请求，消息体示例：

给一批指定的用户发送消息，每个用户发送相同的内容

给一批指定的用户发送消息，每个用户发送相同的内容，内容中使用消息模板

###### 响应

 x-result-code定义

 0：成功

 其它：失败

 响应消息

 x-result-code=0时，返回map类型，包含以下字段

 id：消息唯一标识， string类型

##### 批量发送消息

 调用⽅法： POST {Base URL}/open/mq/messages?channel={channel}

 权限要求： apibase

###### 请求

**表 9**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 参数名称 | 是否必须 | 参数类型 | 参数位置 | 说明 |
| channel | yes | string | query | 发送渠道 |
| messages | yes | Message[] | body | 要发送的消息数组 |

常用的请求，消息体示例

 给一批指定的用户发送消息，每个用户发送相同的内容

给一批指定的用户发送消息，每个用户发送相同的内容，内容中使用消息模板

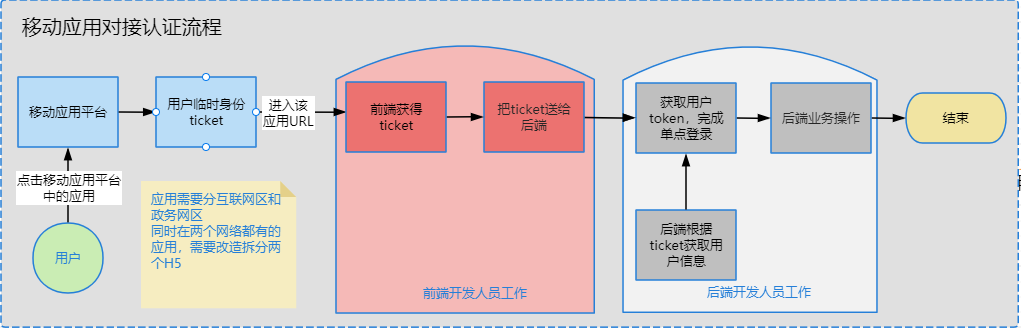
## 统一身份认证对接

用户进入移动应急APP时，已通过账密、证书等方法，完成了身份识别。本文中的统一登录是指：当用户打开移动应用时，通过静默登录移动应用方式，自动识别当前请求的用户身份信息，而无需用户再次登录。

移动应急平台采用票据（ticket）来传递静默登录信息。ticket具有一次性、短时效使用的限制。静默登录通常只在跳转时调用一次，登录成功后，移动应用应使用合适的机制，保存用户的登录状态。

开放平台支持：URL和JS SDK获取票据两种模式，移动应用可根据用户入口类型选择实现方式。如果用户仅从工作台进入，可使用URL模式，此时移动应用可以不引入JS SDK；如果用户在移动应急APP之外打开移动应用，如：点击短信中的链接进入，应采用JS SDK方式。

### 移动应用对接认证流程图



### URL获取模式

当用户从工作台点击移动应用时，工作台会根据应用和用户信息，在移动应用的入口地址参数最后附加票据，移动应用的后台收到请求后，应提取出票据参数，并请求移动应急平台验证票据的合法性。如果验证通过，会返回用户的基础信息，用于匹配移动应用系统的内部用户。[例如：移动应用的URL为](https://app.abc.com/index.html%EF%BC%8C%E7%94%A8%E6%88%B7%E7%82%B9%E5%87%BB%E5%BE%AE%E5%BA%94%E7%94%A8%E5%90%8E%EF%BC%8C%E5%B7%A5%E4%BD%9C%E5%8F%B0%E5%9C%A8%E5%85%A5%E5%8F%A3URL%E5%90%8E%E5%A2%9E%E5%8A%A0%E7%B1%BB%E4%BC%BCa2_ticket=abc.123%E7%9A%84%E5%8F%82%E6%95%B0%EF%BC%8C%E5%AE%9E%E9%99%85%E4%BC%A0%E5%85%A5%E5%BA%94%E7%94%A8%E7%9A%84URL%E4%B8%BA%EF%BC%9Ahttps://app.abc.com/index.html?a2_ticket=abc.123%E3%80%82)<https://app.abc.com/index.html>[，用户点击移动应用后，工作台在入口URL后 增加类似a2\_ticke t=abc.123的参数，实际传入应用的URL为：](https://app.abc.com/index.html%EF%BC%8C%E7%94%A8%E6%88%B7%E7%82%B9%E5%87%BB%E5%BE%AE%E5%BA%94%E7%94%A8%E5%90%8E%EF%BC%8C%E5%B7%A5%E4%BD%9C%E5%8F%B0%E5%9C%A8%E5%85%A5%E5%8F%A3URL%E5%90%8E%E5%A2%9E%E5%8A%A0%E7%B1%BB%E4%BC%BCa2_ticket=abc.123%E7%9A%84%E5%8F%82%E6%95%B0%EF%BC%8C%E5%AE%9E%E9%99%85%E4%BC%A0%E5%85%A5%E5%BA%94%E7%94%A8%E7%9A%84URL%E4%B8%BA%EF%BC%9Ahttps://app.abc.com/index.html?a2_ticket=abc.123%E3%80%82)<https://app.abc.com/index.html?>[a2\_ticket=abc.123。](https://app.abc.com/index.html%EF%BC%8C%E7%94%A8%E6%88%B7%E7%82%B9%E5%87%BB%E5%BE%AE%E5%BA%94%E7%94%A8%E5%90%8E%EF%BC%8C%E5%B7%A5%E4%BD%9C%E5%8F%B0%E5%9C%A8%E5%85%A5%E5%8F%A3URL%E5%90%8E%E5%A2%9E%E5%8A%A0%E7%B1%BB%E4%BC%BCa2_ticket=abc.123%E7%9A%84%E5%8F%82%E6%95%B0%EF%BC%8C%E5%AE%9E%E9%99%85%E4%BC%A0%E5%85%A5%E5%BA%94%E7%94%A8%E7%9A%84URL%E4%B8%BA%EF%BC%9Ahttps://app.abc.com/index.html?a2_ticket=abc.123%E3%80%82)

流程如下：

* 用户点击移动应用，工作台根据应用及用户信息，生成票据，生成拼接票据之后的入口地址
* 工作台打开携带票据参数据入口址
* 移动应用前端或后端从URL中提取出票据，如果是在前端提取，提取后应传给移动应用后端
* 移动应用后端调用移动应急平台“用ticket换取用户信息接口”
* 如果获取用户基础信息成功，即视为登录成功，移动应用系统自行管理用户的登录状态

### JS SDK获取模式

在移动应急APP外（如：短信中的链接）打开移动应用时，无法自动在URL中追加票据参数。移动应用可以调用JS SDK获取票据。

JS SDK中的接口为：ecpot.getTicket(options)

JS SDK兼容URL模式，如果URL中带有a2\_ticket字段，会使用URL的a2\_ticket.

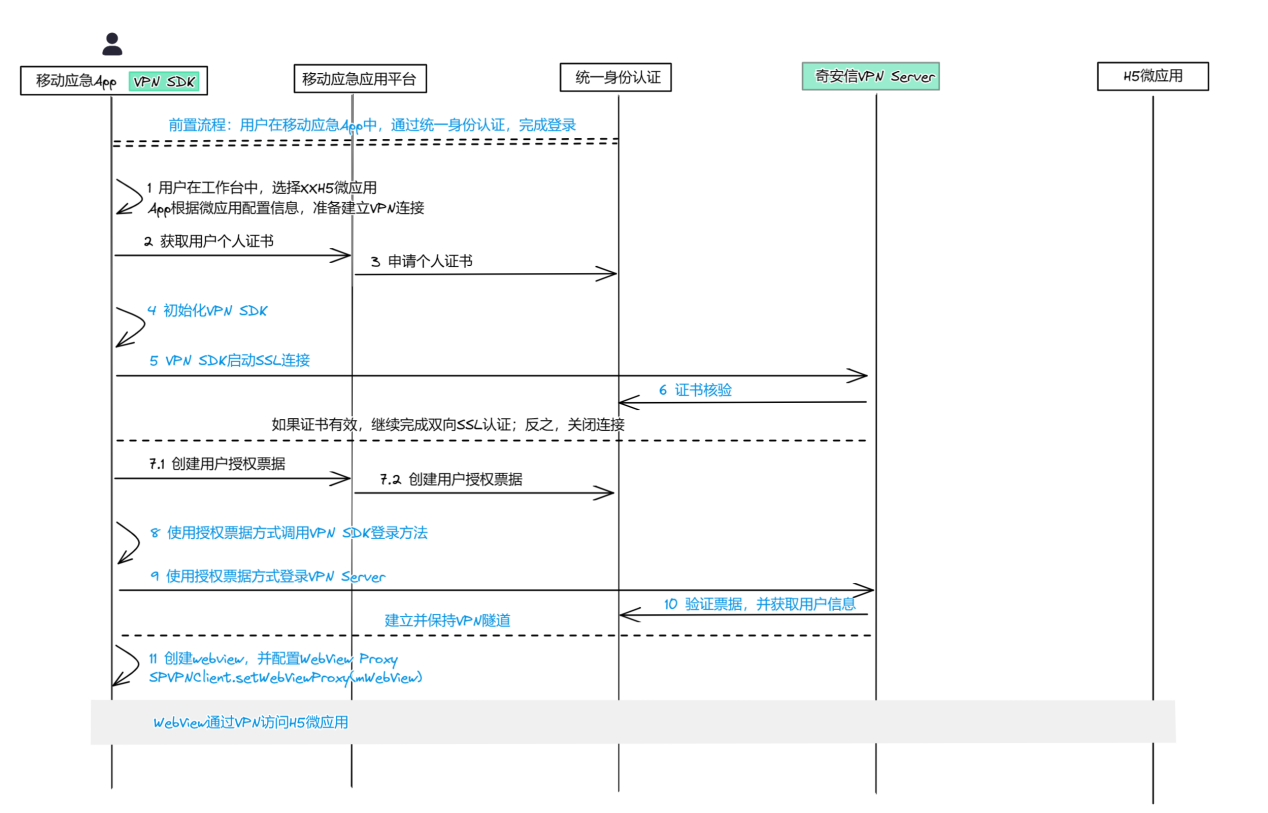
移动应用须引⼊ JS SDK（参考4.6.9章节），流程如下：

* 用户访问移动应用链接，如：点击短信内的链接
* 系统浏览器加载移动应用，移动应用先按照自有逻辑判断用户是否已登录。如果已登录，登录流程结束
* 如果未登录移动应用系统，调用ecpot.getTicket
* 跳转至统一登录页
* 统一登录页自动检查用户是否已登录，如果尚未登录，展示登录窗口（将暂存第三方系统跳转的appId、redirectUrl）。直至用户登录成功，才继续下一步流程
* 统一登录页根据应用及用户信息，生成票据
* 统一登录页根据redirectUrl+ticket，生成重定向地址(在redirectUrl后追加a2\_ticket=xxxx)，导航至该地址
* 移动应用系统通过ecpot.getTicket接口从URL中提取票据
* 移动应用后端调用移动应急平台“用ticket换取用户信息接口”
* 如果获取用户基础信息成功，即视为登录成功，移动应用系统自行管理登录状态；如果失败，调用ecpot.getTicket重定向回统一登录

## VPN服务调用

移动应急APP负责对接VPN服务，H5移动应用申请上架时，应提供网络环境信息及审批信息。在移动应急APP工作台打开H5移动应用时，会根据H5移动应用系统的部署环境，按需建立VPN连接。

VPN服务调用流程图：

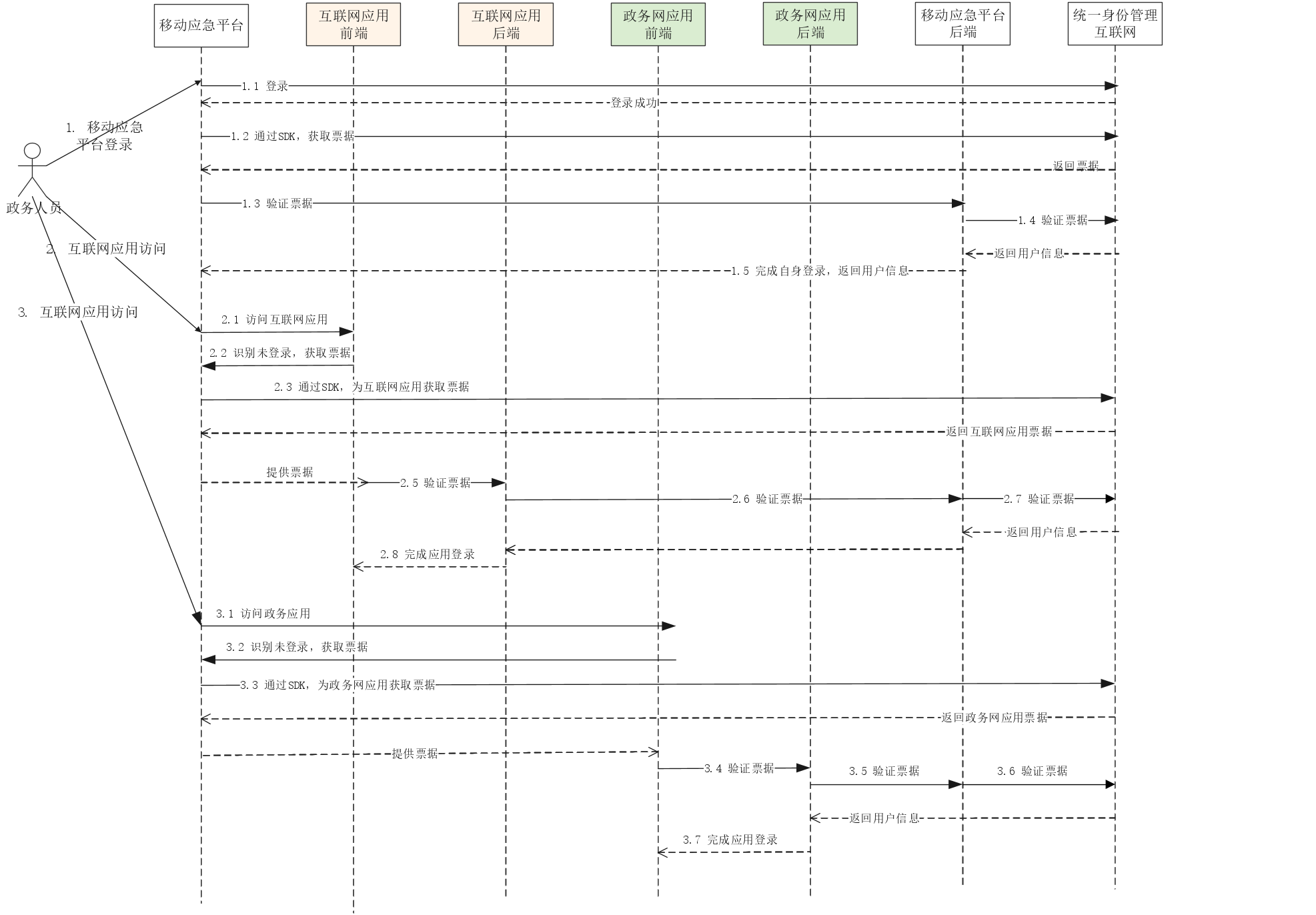


# 部政务用户统一身份认证

部政务用户统一身份认证系统用于实现组织机构、用户、应用数据的集中管理功能，为移动应用系统提供政务用户的统一身份管理、统一身份认证、单点登录、统一授权等身份服务。部政务用户统一身份认证系统的SDK（以下简称SDK）用于移动端应用接入，仅需关注应用业务，即可实现基于数字证书的身份认证、单点登录功能。

## 统一身份认证场景

统一身份认证场景如下图：



1. **认证时序图**

### 政务人员进行移动应急平台登录

* 1. 用户输入身份信息提交互联网侧统一身份管理系统验证通过后完成登录；
  2. 移动应急平台通过认证SDK，从互联网侧统一身份管理系统获取平台自身的登录票据；
  3. 移动应急平台获得自身票据后，调用后端服务进行核验；
  4. 移动应急平台后端提交票据到互联网侧统一身份管理系统进行核验；
  5. 核验通过后，移动应急平台完成自身登录；

### 政务人员访问互联网应用：

* 1. 用户点击互联网应用图标进行访问；
  2. 应用后端识别用户未登录，需要进行登录。应用前端从移动应急平台请求票据；
  3. 移动应急平台通过认证SDK为应用获取票据，并返回给应用；
  4. 应用前端将票据提交至后端进行验证；
  5. 互联网应用后端提交票据到移动应急平台进行票据核验
  6. 移动应急平台后端服务提交票据到互联网侧统一身份管理系统进行核验；
  7. 统一身份管理系统票据核验通过后，移动应急平台返回核验结果给互联网侧应用，互联网应用完成自身登录，可继续进行业务操作；

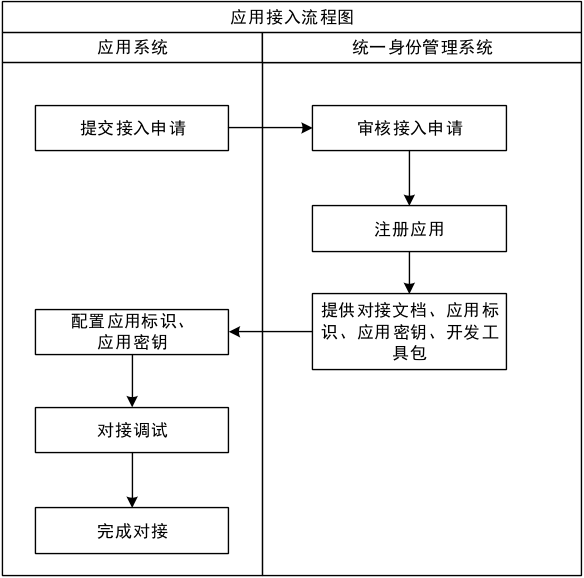
### 政务人员访问政务网应用

* 1. 用户点击政务网应用图标进行访问；
  2. 应用后端识别用户未登录，需要进行登录。应用前端从移动应急平台请求票据；
  3. 移动应急平台通过认证SDK为应用获取票据，并返回给应用；
  4. 应用前端将票据提交至后端进行验证；
  5. 政务网应用后端提交票据至移动应急平台进行票据核验
  6. 移动应急平台后端服务提交票据到互联网侧统一身份管理系统进行核验；
  7. 统一身份管理系统票据核验通过后，移动应急平台返回核验结果给政务网侧应用，政务网应用完成自身登录，可继续进行业务操作。

## 应用系统对接流程

应用系统进行统一身份认证对接前，需先在测试环境进行集成对接，对接完成后，再在正式系统进行上线。

对接基本流程如下：



* 1. 应用系统负责人提交应用信息，申请接入统一身份认证系统；
  2. 统一身份管理系统负责人对应用系统负责人提交的信息进行审批，如果发现问题与应用系统负责人进行沟通；
  3. 统一身份管理系统负责人在统一身份管理系统注册应用，生成应用身份信息，如应用系统标识、应用系统密钥等；
  4. 统一身份管理系统负责人将对接信息提供给应用系统负责人，如对接文档、应用系统标识；
  5. 应用系统开发人员根据对接文档进行对接开发、联调和测试。
  6. 联调测试通过后，统一身份管理系统负责人确认对接结果，确认后按上述流程进行正式环境配置上线。

## 接口描述

### 获取访问票据（SDK调用）

SDK通过该接口获取应用票据。

|  |  |
| --- | --- |
| **URL** | http://sso.example.com/authn-api/v5/authn/sdk/ticket |
| **调用方式** | POST |
| **参数说明** | {    "sessionId": "63f95d5f-4328-42b2-a533-2d8e00441519",//会话ID，必传  "clientId": "", // 非必传参数，当应用为OAUTH协议时传  "appId": "123456789"， //应用ID，非必传，应用Id/应用地址2个传其中一个  "appUrl": "http://10.0.41.10:8030/ssoDemoWithInterface/"//应用地址，非必传，应用Id/应用地址2个传其中一个  } |
| **请求示例** | http://sso.example.com/authn-api/v5/cas/login?service=http://app.example.com |
| **请求头** | Content-Type:application/json; charset=UTF-8 |
| **返回值** | {  "code": "0", // 状态码  "traceId": "xxx", // 请求唯一id，方便追踪日志  "timestamp": 1544154563599, // 服务器时间戳  "result":{  "zone": "1", // 票据所在区域，按应急统一身份系统区域配置获取  "ticket": "ST-63f95d5f-4328-42b2-a533-2d8e00441519", // ST票据  "exprieIn": "60" //票据过期时间，单位：s  }  } |
| **返回值说明** |  |
| **备注** | ticket：服务票据（ST）有效期100秒，且只能只用一次  zone：区域，验证票据的时候需要携带该参数。门户需要将zone信息与ticket信息一并反馈给业务系统。 |

### 验证访问票据（业务系统）

应用系统获取访问票据后，通过该接口验证票据，并获得用户信息。

|  |  |
| --- | --- |
| **URL** | http://sso.example.com/authn-api/v5/cas/{appKey}/p3/serviceValidate |
| **调用方式** | get |
| **参数说明** | **appKey：系统分配的应用appKey**  **service: 应用系统地址**  **ticket：服务票据（ST）**  **zone：区域。获取票据时如果门户返回了该参数，需要一同传递给后端。**  **format：返回数据格式，支持json和xml，默认xml** |
| **请求示例** | http://sso.example.com/authn-api/v5/cas/123456789/p3/serviceValidate?service=http://app.example.com&ticket=ST-123&format=json&zone=1 |
| **请求头** | descript |
| **返回值** | {  "serviceResponse": {  "authenticationSuccess": {  "user": "973509957c4941fcb73fbf3c67e7554a",  "attributes": {  "USER\_ID": "xxxxxxxx",  "IDCARD\_UUID": "xxxxxxxx",  "USER\_NAME": "高波",  "MOBILE": "xxx"  }  }  }  }  失败返回值：  {  "serviceResponse": {  "authenticationFailure": {  "code": "INVALID\_TICKET",  "description": "无效的票据"  }  }  } |
| **返回值说明** |  |
| **备注** | 如果要实现单点退出，则要在会话中保存用户与ST关联关系  示例中只列了关键字段，其他属性参见《政务人员属性信息》 |

### 验证访问票据（门户代理）

移动应急平台门户通过该接口，为各业务系统进行票据验证。

|  |  |
| --- | --- |
| **URL** | http://sso.example.com/authn-api/v5/cas/{appKey}/p3/proxyServiceValidate |
| **调用方式** | GET |
| **参数说明** | appKey：系统分配的应用id，门户appKey  service: 门户应用系统地址  ticket：业务系统服务票据（ST）  zone：区域。获取票据时如果门户返回了该参数，需要一同传递给后端。  format：返回数据格式，支持json和xml，默认xml  bizAppKey : 票据所属的业务系统appKey |
| **请求示例** | http://sso.example.com/authn-api/v5/cas/123456789/p3/proxyServiceValidate?service=http://app.example.com&ticket=ST-123&format=json&zone=1&bizAppKey=987654321 |
| **请求头** | |  |  |  | | --- | --- | --- | | **参数** | **参数说明** | **是否必选** | | once | 随机数，32位，可用UUID表示 | 必选 | | ts | 时间戳 | 必选 | | appKey | 应用标识 | 必选 | | signMethod | 签名方法 | 必选，参见“[签名方法](#_接口签名规则)” | | signData | 参数的签名 | 必选 | |
| **返回值** | {  "serviceResponse": {  "authenticationSuccess": {  "user": "973509957c4941fcb73fbf3c67e7554a",  "attributes": {  "USER\_ID": "xxxxxxxx",  "IDCARD\_UUID": "xxxxxxxx",  "USER\_NAME": "高波",  "MOBILE": "xxx"  }  }  }  }  失败返回值：  {  "serviceResponse": {  "authenticationFailure": {  "code": "INVALID\_TICKET",  "description": "无效的票据"  }  }  } |
| **返回值说明** |  |
| **备注** | 示例中只列了关键字段，其他属性参见《政务人员属性信息》 |

### 统一认证登出（SDK调用）

应用系统退出登录时，跳转至该地址，完成统一身份认证系统的退出。

|  |  |
| --- | --- |
| **URL** | http://sso.example.com/authn-api/v5/cas/logout?service=http://app.example.com |
| **调用方式** | get |
| **参数** | service：应用系统指定的登出地址（一般为应用登录首页） |
| **请求示例** | http://sso.example.com/authn-api/v5/cas/logout |
| **请求头** | 无 |
| **返回值** | 跳转至登录页面 |
| **返回值说明** |  |
| **备注** | 应用系统发出登出请求给统一身份认证系统，统一身份认证系统收到请求后会通知所有系统登出此用户 |

### 应用登出通知（应用系统提供）

当用户登出时，认证系统会给所有本次单点会话相关的应用系统发送消息，通知用户登出事件，应用系统需要按要求终止用户登录会话。

|  |  |
| --- | --- |
| **URL** | （应用系统提供地址） |
| **调用方式** | post |
| **请求体** | <samlp:LogoutRequest  xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"  xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"  ID="[RANDOM ID]"  Version="2.0"  IssueInstant="[CURRENT DATE/TIME]">  <saml:NameID>@NOT\_USED@</saml:NameID>  <samlp:SessionIndex>[ST]</samlp:SessionIndex>  </samlp:LogoutRequest> |
| **参数说明** | ID：随机数  IssueInstant：当前时间戳  ST: 应用系统登录时获取的票据 |
| **请求示例** | http://sso.example.com/sso/logout |
| **请求头** | Content-Type:application/x-www-form-urlencoded |
| **返回值** |  |
| **返回值说明** |  |
| **备注** | 要实现单点登出需要将用户与ST进行关联，从而实现对指定用户的登出  注意：请求体为标准表单请求，不是JSON格式 |

## 认证SDK集成

### 接入模式及要求

#### 客户端要求

客户端集成时，需要满足以下前提条件：

1. **系统版本要求**

1) Android系统支持5.0及以上版本

1. **设备唯一标识**
2. Android客户端有以下方案

ANDROID\_ID(Android8及以上版本)

IMEI(Android8以下版本)

Wlan address

Bluetooth address

SIM ICCID

#### 依赖说明

// 证书管理SDK

implementation(name: 'zta-client-Android-SMF-xxx', ext: 'aar')

// 认证SDK

implementation(name: 'zta-client-Android-Uikit-xxx', ext: 'aar')

// 调用okhttp 需要

implementation 'org.jetbrains.kotlinx:kotlinx-coroutines-android:1.4.2'

// kztSdk依赖

implementation 'com.google.code.gson:gson:2.8.7'

implementation 'com.squareup.retrofit2:retrofit:2.9.0'

implementation 'com.squareup.retrofit2:converter-gson:2.9.0'

implementation 'com.squareup.retrofit2:converter-scalars:2.9.0'

implementation 'com.squareup.okio:okio:2.9.0'

// kztSmfSdk依赖

implementation 'org.bouncycastle:bcprov-jdk15on:1.61'

implementation 'org.bouncycastle:bcpkix-jdk15on:1.61'

// sdk内包含了指纹识别

implementation 'androidx.biometric:biometric:1.1.0'

// 当使用国密OKHttp，导致OKHttp冲突时，去除公开版的调用

configurations.all {

    exclude group: 'com.squareup.okhttp3', module: 'okhttp'

    exclude group: 'com.squareup.okhttp3', module: 'logging-interceptor'

}

#### 访问权限

<uses-permission android:name="android.permission.BLUETOOTH" />

<uses-permission android:name="android.permission.BLUETOOTH\_ADMIN" />

<uses-permission android:name="android.permission.BLUETOOTH\_CONNECT" />

<uses-permission android:name="android.permission.CHANGE\_WIFI\_STATE" />

<uses-permission android:name="android.permission.INTERNET" />

<uses-permission android:name="android.permission.NEARBY\_WIFI\_DEVICES" />

<uses-permission

    android:name="android.permission.NETWORK\_SETTINGS"

    tools:ignore="ProtectedPermissions" />

<uses-permission android:name="android.permission.READ\_PHONE\_STATE" />

<uses-permission android:name="android.permission.ACCESS\_NETWORK\_STATE" />

<uses-permission android:name="android.permission.ACCESS\_WIFI\_STATE" />

<uses-permission android:name="android.permission.ACCESS\_FINE\_LOCATION" />

<uses-permission android:name="android.permission.ACCESS\_COARSE\_LOCATION" />

<!--指纹相关依赖-->

<uses-permission android:name="android.permission.USE\_FINGERPRINT" />

<uses-permission android:name="android.permission.USE\_BIOMETRIC" />

<!--UIKIT组件需要权限-->

<uses-permission android:name="android.permission.WRITE\_EXTERNAL\_STORAGE" />

<uses-permission android:name="android.permission.READ\_EXTERNAL\_STORAGE" />

<uses-permission android:name="android.permission.MANAGE\_EXTERNAL\_STORAGE" />

<uses-permission android:name="android.permission.CAMERA" />

### 接入模式

#### SDK接口

对身份认证业务较熟悉的开发人员可采用SDK接口接入方式，通过[身份认证接口](#_身份认证接口)实现身份认证功能。在身份认证采用CAS协议实现，需要通过证书实现认证时，APP可考虑使用[本地用证接口](#_本地用证接口)使用证书。

如果APP对应服务端集成了CDS在线发证服务，可使用[在线发证接口](#_在线发证接口)实现在线发证。

有网络通道加密需求的业务，可使用SSL接口实现。

#### JSBridge接口

针对独立的Web应用实现身份认证业务的需求，SDK支持JS脚本函数。JS-SDK是对SDK接口的JS封装，Web应用通过统一的[JS-SDK接口](#_JS-SDK接口)实现身份认证需求。

**实现方式**

// Application 中初始化

KZTSdk kztSdk = KZTSdk.getInstance();

kztSdk.initSdk(app, "", logPath, appKey, appSecret, deviceInfo);

kztSdk.setOkhttpBuilder(gmBuilder);

ResultBean<String> resultBean = kztSdk.initConfigServer(configServerUrl);

// Webview界面中

WebSettings settings = webView.getSettings();

String userAgent = settings.getUserAgentString();

settings.setUserAgentString(userAgent + " koal");

JsSdkManager helper = KztUikitSdk.getKztSdk().setupJavascriptBridge(mActivity, webView);

helper.registerUiasFunc();

helper.registerCertFunc();

#### UIKit组件(推荐)

针对不熟悉身份认证业务的开发者，SDK支持以[UIKit组件](#_UIKit接口)集成的方式集成身份认证功能。UIKit是对身份认证业务的UI交互封装，APP集成接口少，功能完善。为了满足APP对于界面风格和交互特点的特异性，UIKit支持UI界面自定义和交互方式扩展。

#### uni-app插件

针对uni-app应用实现身份认证业务的需求，SDK支持uni-app插件功能。uni-app插件是对SDK接口的封装，uni-app应用通过统一的uni-app插件实现身份认证需求。

### SDK接口说明

#### SDK接口

SDK接口是针对各模块接口的简单封装，包括身份认证、在线发证、本地证书管理模块。均以同步接口提供，应用层使用时，根据实际情况通过子线程调用。

#### KZTSdk接口

##### 初始化

**public** **void** initSdk(Application app, String runPath, String logPath, String appKey, String appSecret, DeviceInfo deviceInfo)

##### 设置国密OKHTTP请求

/\*\*

 \* 设置okhttp请求方式，可以放在 application 调用

 \*

 \* @param gmBuilder okhttp 请求实现

 \* @return

 \*/

**public** **void** setOkhttpBuilder(OkHttpClient.Builder gmBuilder)

#### 身份认证接口

身份认证接口是对IDaaS服务接口的简单封装，APP可通过此模块接口实现身份认证功能。

提供身份认证接口的文件：KZTUiasMgr.java

##### 初始化

初始化接口用于客户端获取服务登录策略，策略包含登录方式集合、认证上下文、自动登录票据参数名等信息。APP应先调用该接口创建身份认证实例对象(后续使用该实例对象调用其他认证接口)，并通过该接口返回的登录方式集合显示登录界面。

/\*\*

 \* 初始化配置服务, 通过配置地址获取各个服务地址并初始化

 \*

 \* @param configServerUrl 配置服务地址

 \* @return

 \*/

**public** void initConfigServer(String configServerUrl, KztCallback<Boolean> callback)

##### 用户登录

初始化成功后，根据初始化接口返回的登录方式集合，默认显示首个登录方式界面，如果登录方式集合有多个，支持登录方式切换。

若需要多因素登录，登录接口会响应后续登录方式集合和对应账号相关信息，APP根据响应跳转到其他登录界面继续执行登录流程。多因素登录一般会执行两种或两种以上的登录过程才会生成用户票据，并返回用户信息。

目前支持的登录方式有：

pwd：密码登录

pki\_cert：签名验签登录([先获取签名随机数](#_获取签名随机数))

ssl\_cert：安全认证网关认证

id\_number：身份证件登录

email\_code：邮箱验证码登录([先获取邮箱验证码](#_发送邮箱验证码))

dynamic\_token：OTP动态码登录(可配合格尔手机令牌使用)

mobile\_code：短信验证码登录([先发送短信验证码](#_发送手机验证码))

fido\_uaf：生物特征登录([先注册FIDO协议](#_FIDO注册))

/\*\*

 \* 口令登录

 \*/

**public** SsoResultBean<NextAuthBean> loginByPwd(String userName, String passWord)

/\*\*

 \* 短信登录

 \*/

**public** SsoResultBean<NextAuthBean> loginBySms(String mobile,  String code)

/\*\*

 \* 通用登录接口

 \*

 \* @param authInfo {"authnType": "以下加密方式",

 \*  pwd:{"passWord": "","userName": ""}

 \*  pki\_cert:{"userPin":@"123456"}

 \*  ssl\_cert:{"userPin":@"123456"}

 \* mobile\_code:{"mobile": "","code": ""}

 \* qr\_code:{"qrId": ""}

 \*  fido:{"authenticationResponses": ""}

 \*  face:{"faceData": "","username": ""}

 \*  dingtalk\_qr:{"code": ""}}

 \* @return

 \*/

**public** SsoResultBean<NextAuthBean> login(JSONObject authInfo)

##### 持续登录

用户登录成功后，为H5应用登录获取自动登录票据时，若该H5应用配置的认证链分数比已产生的会话分数高，需要执行持续认证。获取自动登录票据接口会响应需要持续认证的认证方式集合，APP应该根据该响应显示持续认证登录界面，调用该接口执行持续认证。

持续认证与用户登录一样，会有多因素认证的情况，认证链与用户登录流程一致。

/\*\*

 \* 持续登录

 \*

 \* @param serviceUrl 需要登录的url

 \* @param authInfo   {"authnType": "", // 以下登录类型

 \*                   "cert": { // pki证书登录

 \*                   "randomNo": "",

 \*                   "signCert": "",

 \*                   "signData": "",

 \*                   "verifyRandomNo": true

 \*                   },

 \*                   "dingtalkQr": { // 钉钉扫码认证

 \*                   "code": "" 临时授权码

 \*                   },

 \*                   "emailCode": { // 邮箱验证码认证

 \*                   "code": "", 邮箱验证码

 \*                   "email": "" 邮箱

 \*                   },

 \*                   "face": {  人脸认证

 \*                   "faceData": "",  人脸数据

 \*                   "username": "" 用户名

 \*                   },

 \*                   "fidoUaf": { FIDO\_UAF认证

 \*                   "authenticationResponses": "" 认证响应数组字符串

 \*                   },

 \*                   "idnumber": { 证件编号认证

 \*                   "number": "", 证件编号

 \*                   "numberEncrypt": "", 证件编号加密方式（PLAIN：明文;SHA-1：sha1摘要;SHA1+SM3：sha1+sm3摘要）

 \*                   "userName": "" 用户名称

 \*                   },

 \*                   "mobileCode": { 手机验证码认证

 \*                   "code": "", 手机验证码

 \*                   "mobile": "" 手机号码

 \*                   },

 \*                   "pwd": { 账号密码认证

 \*                   "passWord": "", 密码

 \*                   "pwdEncrypt": "", 密码加密方式（PLAIN：明文;SHA-1：sha1摘要;SHA1+SM3：sha1+sm3摘要）

 \*                   "userName": "" 用户名称

 \*                   },

 \*                   "qrCode": { 手机扫码认证

 \*                   "qrId": "" 二维码ID

 \*                   },

 \*                   "qywxOauth": { 企业微信oauth认证

 \*                   "code": "" 临时授权码

 \*                   },

 \*                   "sslCert": { 网关证书认证

 \*                   "certUniqueId": "", 证书唯一标识

 \*                   "userUniqueId": "" 用户唯一标识

 \*                   }

 \*                   }

 \* @return

 \*/

**public** SsoResultBean<String> continueLogin(String serviceUrl, JSONObject authInfo)

;

##### 获取签名随机数

登录方式为证书签名验签登录时，需要先从服务获取签名随机数，然后通过证书签名接口签名后提交登录。

/\*\*

 \* 获取随机数

 \* @return

 \*/

**public** SsoResultBean<String> getRandom()

##### 发送手机验证码

登录方式为短信登录时，需要先发送手机验证码，然后提交登录。

/\*\*

 \* 获取短信验证码

 \* @param phoneNum 接收短信的手机号

 \* @return

 \*/

**public** SsoResultBean<Boolean> getMobileCode(String phoneNum)

##### 发送邮箱验证码

登录方式为邮箱登录时，需要先发送邮箱验证码，然后提交登录。

/\*\*

 \* 发送短信验证码

 \*

 \* @param email

 \* @return

 \*/

**public** SsoResultBean<Boolean> getEmailCode(String email)

##### 获取应用票据

用户登录成功后，通过此接口获取票据用于APP登录。

/\*\*

 \* 获取应用票据

 \* @param clientId oAuth时填写 默认未空

 \* @return

 \*/

**public** SsoResultBean<String> getLoginTicket(String clientId)

##### 获取单点登录票据

用户登录成功后，通过此接口获取票据用于H5应用单点登录。

/\*\*

 \*

  \* 获取应用登录票据

 \*

 \* @param serviceUrl H5应用url

 \* @return

 \*/

**public** SsoResultBean<String> getAutoLoginTicket(String serviceUrl)

##### 会话续期

/\*\*

 \* 会话延期

 \*

 \* @return

 \*/

**public** SsoResultBean<String> renewSession()

##### 注销会话

**public** SsoResultBean<Boolean> logout()

#### 证书管理接口

##### 获取对应接口类

// 获取证书管理接口类

ICertHelperApi temp = CertApi.getInstance().getCertHelperImpl();

// 获取设备管理接口类

IDeviceMgr deviceMgrTemp = CertApi.getInstance().getDeviceMgrImpl();

##### 证书助手接口

###### 初始化

当需要在线发证、协同签名、证书续期、证书废除时，调用ICertHelperApi 证书助手类

/\*\*

\* 初始化证书助手

\*

\* @param userID  当前用户ID，无则传当前设备ID

\* @param cdsServerUrl cds发证服务器地址

\* @param sksServerUrl 协同签名服务器地址

\* @param publicIP     公网IP,无则传空

\* @param dataPath      数据存储目录，默认传空

\* @return

\*/

KZTSdk init(String userId, String cdsServerUrl, String sksServerUrl, String publicIp, String dataPath)

###### 获取设备管理器

IDeviceMgr getDeviceMgr();

###### 查询证书状态

/\*\*

\* 查询证书状态

\*

\* @return

\*/

ResultBean queryCertOperateState();

###### 快速发证

/\*\*

 \* 签发证书，快速发证模式

 \*

 \* @param userName 用户名

 \* @param pwd      密码

 \* @param pin      证书pin码

 \* @return

 \*/

ResultBean certIssuer(String userName, String pwd, String pin);

###### 废除证书

/\*\*

 \* 废除证书

 \*

 \* @param userName 用户名

 \* @param pwd      密码

 \* @param certCid  证书cid编号

 \* @return

 \*/

ResultBean revokeCert(String userName, String pwd, String certCid);

##### 设备管理接口

当需要导入证书和使用证书时可以调用 IDeviceMgr 设备管理类

###### 初始化

/\*\*

\* 初始化设备管理器

\*

\* @param userId 当前用户ID，无则传当前设备ID

\* @param path   sdk 路径, 默认传空

\*/

**void** init(String userId, String path);

###### 激活当前设备

/\*\*

 \* 激活当前设备

 \*

 \* @param bean 当前设备信息

 \*/

**void** activeDevice(CertDeviceBean.StrJsonBean bean);

###### 验证pin码

/\*\*

 \* 验证pin码

 \*

 \* @param pin 证书密码

 \* @return 成功/失败

 \*/

ResultBean verifyPin(String pin);

###### 导入pfx证书

/\*\*

 \* 导入证书

 \*

 \* @param certPath pfx证书的base64

 \* @param pwd      证书密码

 \* @return

 \*/

ResultBean importPfxCert(String certPath, String pwd);

###### 导出公钥

/\*\*

 \* 导出公钥

 \*

 \* @param flagType 0 加密公钥 1 签名公钥

 \* @return 公钥base64

 \*/

ResultBean<**byte**[]> exportPublicKey(**int** flagType);

###### 导出证书

/\*\*

 \* @param certType 证书类型 1表示签名证书，0表示加密证书

 \* @return

 \*/

ResultBean<**byte**[]> exportCert(**int** certType);

###### pkcs#1签名/数字签名

/\*\*

 \* 数字签名

 \*

 \* @param src    签名原文

 \* @param type   签名类型  1 表示 PM-BD 签名,2 表示 SM2/RSA 签名,3 SSL 建链定制签名

 \* @param mdType 摘要类型，1 表示 MD5,2 表示 SHA1,3 表 示 SM3 ， 4 表 示 SHA256

 \* @return 签名后的base64数据

 \*/

ResultBean<**byte**[]> signData(**byte**[] src, String type, String mdType);

###### 验证数字签名

/\*\*

 \* 验证数字签名

 \*

 \* @param b64OriginData 原文base64

 \* @param b64SignData   签名值base64

 \* @return

 \*/

ResultBean<String> verifyData(String b64OriginData, String b64SignData);

###### pkcs#7签名

/\*\*

 \* p7数字信封签名

 \*

 \* @param random   签名原文

 \* @param signHash 非国密  SHA1   国密  SM3

 \* @param attached 是否添加原文 0 包含原文和证书 1 不包含原文 2 不包含证书 3 不包含原文和证书

 \* @return

 \*/

ResultBean<String> signMessage(String random, String signHash, **int** attached);

###### 验证pkcs#7签名

/\*\*

 \* 验证数字信封

 \*

 \* @param b64OriginData   签名原文base64

 \* @param b64Pkcs7Message 签名结果base64

 \* @return

 \*/

ResultBean<String> verifyMessage(String b64OriginData, String b64Pkcs7Message);

###### 获取证书列表

/\*\*

 \* 获取证书列表

 \*

 \* @return

 \*/

ResultBean<List<CertInfoBean>> getCertList();

###### 删除本地证书

/\*\*

 \* 删除本地证书

 \*

 \* @return

 \*/

ResultBean delLocalCert();

###### 生成密钥对

/\*\*

 \* 创建秘钥对, 方法内部创建应用

 \*

 \* @param keyType 0 sm2 1 RSA

 \* @param pin     证书密码

 \* @param alias   证书别名

 \* @return 公钥base64

 \*/

ResultBean<**byte**[]> genKeypair(String keyType, String pin, String alias);

#### 二维码授权接口

##### 二维码扫码

/\*\*

 \*

  \* 获取登录二维码信息

 \*

 \* @param qrId 二维码

 \* @return

 \*/

**public** SsoResultBean<ScanQrResultBean> getQrInfo(String qrId)

##### 二维码授权

/\*\*

 \*

 \* 授权二维码登录

 \*

 \* @param qrId   二维码信息

 \* @param random 二维码信息中的随机数

 \* @return

 \*/

**public** SsoResultBean<Boolean> confirmQrInfo(String qrId, String random)

#### FIDO管理接口

##### FIDO注册

/\*\*

 \* 生物特征(Fido)注册

 \* @param activity  上下文对象

 \* @param userName  用户id

 \* @throws Exception

 \*/

**public** **void** register(Activity activity, String userName) **throws** Exception

##### FIDO认证

/\*\*

 \* 生物特征认证

 \* @param activity

 \*/

**public** **void** auth(Activity activity)

##### FIDO注销

/\*\*

 \* 注销生物特征

 \* @param activity

 \*/

**public** **void** unregister(Activity activity)

#### UIKit组件

UIKit是通过交互界面对原子接口封装后的组件，APP可直接通过UIKit组件接入身份认证功能，而无需关心具体的认证业务流程。UIKit组件具有极简接口，丰富认证方式，并支持APP对UIKit交互界面进行自定义等特点。

提供UIKit接口的文件**KztUikitSdk.java**

##### 初始化

UIKit组件初始化，缓存文件路径。APP通过初始化的实例对象调用相应接口。

/\*\*

 \* 初始化koal sdk，需要在Application 中调用

 \*

 \* @param app

 \* @param appKey     应用id，

 \* @param appSecret  应用秘钥

 \* @param deviceInfo 设备信息

 \* @param logPath    日志地址，为空时/sdcard/Android/data/应用包名/files/sdk/log

 \* @param useCrash   是否启用崩溃记录 默认为 false

 \* @return

 \*/

**boolean** init(Application app, String appKey, String appSecret, DeviceInfo deviceInfo, String logPath, **boolean** useCrash);

##### 设置项目配置

UIKIT SDK中使用默认配置，当需要修应用名称、图标，或隐藏某个功能时，可以通过设置项目配置进行修改。其他配置参考[TeamConfig](#_TeamConfig);

TeamConfig teamConfig = **new** TeamConfig();

// 登录界面应用图标

teamConfig.setAppIcon(R.mipmap.icon\_icon);

// 登录界面应用名

teamConfig.setAppName("测试标题");

// 是否显示配置服务器

teamConfig.setShowConfigServer(**false**);

##### 设置证书助手

UIKit组件初始化后，需要通过setCertHelperImpl接口配置证书助手对象，设置[(ICertHelperApi)](#_获取对应接口类)实现类，实现发证功能接口调用。

/\*\*

\* 设置具体的证书助手实现类

\* @param certHelperImpl

\*/

**void** setCertHelperImpl(ICertHelperApi certHelperImpl);

##### 设置国密请求

SDK默认使用标准请求方式，当需要通过国密协议请求时，通过内置SMF SDK获取对应的国密对象进行设置，从而灵活配置请求方式。

OkHttpClient.Builder gmBuilder = KoalOkHttpClientBuilderFactory.getInstance(**false**);

gmBuilder.connectionSpecs(Arrays.asList(

**new** ConnectionSpec

                .Builder(ConnectionSpec.MODERN\_TLS)

                .supportsTlsExtensions(**false**)

                .allEnabledCipherSuites()

                .tlsVersions(TlsVersion.GMVPNv1\_1)

                .build(),

        ConnectionSpec.CLEARTEXT

));

/\*\*

 \* 设置okhttp请求方式，可以放在 application 调用

 \*

 \* @param gmBuilder okhttp 请求实现

 \* @return

 \*/

**public** **void** setOkhttpBuilder(OkHttpClient.Builder gmBuilder)

##### 认证模块初始化

初始化认证模块，通过配置服务地址获取具体的业务地址。

/\*\*

 \* 初始化服务器信息

 \* @param context        上下文对象

 \* @param configServerUrl 配置服务器地址，通过此地址获取详细服务地址

 \* @param callback          接收处理结果

 \* @return

 \*/

void initConfigServer(Context context,String configServerUrl, KztCallback<Boolean> callback);

##### 登录

APP通过初始化的实例对象调用该接口，调起登录界面UI。用户通过与UI的交互执行身份认证流程，直至登录成功后登录界面UI销毁，登录成功后，将返回对应APPID的票据。需要在认证模块初始化callback中调用。

/\*\*

 \* 登录,返回应用票据

 \*

 \* @param context           上下文对象

 \* @param appId             需要获取票据的应用id，由服务端添加后提供，如无需登录后获取票据，则传null

 \* @param callback          接收处理结果

 \* @param changePwdCallback 是否需要修改密码弹窗

 \*/

**void** login(Context context, String appId, KztCallback<TicketInfoBean> callback, KztCallback<Void> changePwdCallback);

##### 获取应用票据

/\*\*

 \* 获取应用服务票据接口

 \*

 \* @param appId    需要获取票据的应用id，由服务端添加后提供

 \* @param callback

 \*/

**void** getLoginTicket(String appId, KztCallback<TicketInfoBean> callback);

##### 获取H5应用自动登录票据

/\*\*

 \* 获取h5应用自动登录票据

 \*

 \* @param serviceUrl H5应用url

 \* @return

 \*/

**void** getAutoLoginTicket(String serviceUrl, KztCallback<TicketInfoBean> callback);

##### 打开提示修改密码弹窗

登录成功后，SDK中自动调用了检查密码有效期策略接口，调用完成后，将在login的changePwdCallback中返回，收到回调后，可以调用此接口，若需要提示修改密码，SDK将弹出密码修改提示窗口，若未弹出，则说明该账号不需要修改密码。

/\*\*

  \* 打开修改密码弹窗

  \*

  \* @param context

  \*/

**void** showChangePwdDialog(Context context)

##### 打开修改密码界面

/\*\*

 \* 打开修改密码界面

 \*

 \* @param context

 \*/

**void** showChangePwdActivity(Context context);

##### 打开账号与安全界面

/\*\*

 \* 打开账号与安全界面

 \*

 \* @param context

 \*/

**void** showAccountSecurityActivity(Context context);

##### 打开我的设备界面

/\*\*

 \* 打开我的设备界面

 \*

 \* @param context

 \*/

**void** showMyDeviceInfoActivity(Context context);

##### 退出登录

通过UIKit形式集成身份认证的APP，需要退出登录时调用此接口。

/\*\*

  \* 退出登录接口

 \*

 \* @param callback

 \*/

**public** **void** logout(KztCallback callback)

#### 界面自定义

##### 注册自定义界面

/\*\*

 \* 添加替换界面

 \*

 \* @param oldActivity     被替换的界面，从RepalceActivityEvent中获取替换界面

 \* @param newActivityName 替换的界面名，例如：com.ab.ab.AbActivity

 \*/

**void** registerInterface(KztUikitUtils.ReplaceActivityEvent oldActivity, String newActivityName);

##### 取消注册自定义界面

/\*\*

 \* 删除替换的界面

 \*

 \* @param oldActivity 删除的界面，从RepalceActivityEvent中获取删除的界面

 \*/

**void** unRegisterInterface(KztUikitUtils.ReplaceActivityEvent oldActivity);

##### 清空注册的自定义界面

/\*\*

 \* 清空所有自定义界面

 \*/

**void** clearRegisterInterface();

##### 可自定义的界面类型

**public** **enum** ReplaceActivityEvent {

       LoginByCdsActivity,            //cds 口令登录

       LoginByCdsSmsActivity,            //cds 短信登录

       LoginActivity, // 登录界面Activity

       MainActivity,                       //主界面

       VerifyPhoneActivity, // 验证手机号

       OtpBindVerifyActivity, // otp绑定认证界面

       OtpSecondVerifyActivity, // otp二次认证

       VerifySmsActivity, // 验证短信

       SettingActivity, // 设置界面

       DeviceDetailActivity, // 证书管理

       CertManagerActivity, // 证书管理

       OAuthActivity, // 提供第三方认证界面

       ChangePwdActivity, // 修改密码窗口

       ChangePwdDialogActivity, // 修改密码提示窗口

       CommonActivity, // 公共界面，内部调用fragment

       AccountSecurityActivity, // 账号与安全界面

       AboutActivity, // 关于界面

       ActivationActivity, // 激活界面

       AuthenticationActivity, // 认证方式管理

       ;

   }

#### uni-app插件

##### 依赖说明

插件配置：

在Assets目录下dcloud\_uniplugins.json文件中添加

|  |
| --- |
| // 。。。 // 其他插件  {  "type": "module",  "name": "KZTSdkPlugin",  "class": "com.koal.uniapp.KZTSdkPlugin"  } |

项目gradle依赖配置：

configurations.all {

    // 使用支持国密ssl的okhttp, 去除公版 (详情见http://git.koal.com/androidgroup/lib-project/okhttp-koal)

    exclude group: 'com.squareup.okhttp3', module: 'okhttp'

    // 优化日志输出,避免输出上传或下载过程中的过大尺寸的body, 去除公版  (详情见http://git.koal.com/androidgroup/lib-project/okhttp-koal)

    exclude group: 'com.squareup.okhttp3', module: 'logging-interceptor'

}

dependencies {

    implementation fileTree(dir: 'libs', include: ['\*.jar'])

    implementation fileTree(dir: 'libs', include: ['\*.aar'])

    // 其他依赖 。。。

    // 零信任SDK插件需要

    implementation 'com.google.code.gson:gson:2.8.7'

    implementation "org.bouncycastle:bcprov-jdk15to18:1.70"

    implementation "org.bouncycastle:bcpkix-jdk15to18:1.70"

    implementation 'org.jetbrains.kotlinx:kotlinx-coroutines-android:1.4.2'

    implementation "com.squareup.okio:okio:2.9.0"

    implementation "com.squareup.retrofit2:retrofit:2.9.0"

    implementation "com.squareup.retrofit2:converter-gson:2.9.0"

    implementation "com.squareup.retrofit2:converter-scalars:2.9.0"

}

##### 初始化SDK

// 获取 kztSDK module

**var** kztSdkModule = uni.requireNativePlugin("KZTSdkPlugin")

/\*\*

 \* 初始化SDK，非耗时

 \*/

initSdk() {

**const** res = kztSdkModule.initSdk({

        // 应用KEY

        'appKey': '28129b05dc454d72aa7fa180f79eab53',

        // 应用密钥

        'appSecret': 'd91954dfc3d84e0daaf4a3ee02280b07'

    })

    // {"error":"","code":"0"}

    console.log('initSdk' + JSON.stringify(res))

    modal.toast({

        message: res,

        duration: 1.5

    });

},

##### 初始化配置服务

initConfigServer() {

    kztSdkModule.initConfigServer({

        // 配置服务地址

        'serverUrl': 'http://10.0.248.151'

    }, (res) => {

        // 结果不用关心

        console.log("initConfigServer" + JSON.stringify(res))

        modal.toast({

            message: res,

            duration: 1.5

        });

    });

}

##### 初始化认证

/\*\*

 \* 初始化认证

 \*/

initAuth() {

    kztSdkModule.initAuth((res) => {

        console.log("initAuth " + JSON.stringify(res))

// {"code":"0","result":{"nextAuthTypes":["pki\_cert","qr\_code"],"autoTicketKey":"aTicket","showBackButton":true,"enhancedAuth":false,"nextAuthTypeConfigMap":{"qr\_code":{"enabled":true,"name":"扫码认证","authnType":"qr\_code","mode":"0"},"pki\_cert":{"signFormat":"PKCS7","enabled":true,"needCert":false,"name":"证书认证","authnType":"pki\_cert","icon":"","neeOriginal":false,"mode":"0"}},"authnContextId":"52605a4e087346f2b0335604c1a5bd1e"}}

        modal.toast({

            message: res,

            duration: 1.5

        });

    });

}

##### 获取本地证书状态

查看所有[证书状态](#_证书状态)

/\*\*

 \* 获取本地证书状态

 \*/

getCertStatus() {

    kztSdkModule.getCertStatus(res => {

        // {"code":"0","result":0}   0 证书不存在 1 证书存在 其他值查看文档

        console.log("getCertStatus " + JSON.stringify(res))

        modal.toast({

            message: res,

            duration: 1.5

        });

    });

}

##### 发证服务发送短信

/\*\*

 \* 发证服务获取短信验证码

 \*/

sendAuthSms() {

    let phoneNum = **this**.phoneNum; // 手机号

    kztSdkModule.sendAuthSms(phoneNum, res => {

        // {"error":"","code":"0"}

        console.log("sendAuthSms" + JSON.stringify(res))

        modal.toast({

            message: res,

            duration: 1.5

        });

    });

}

##### 发证服务通过短信登录

/\*\*

 \* 通过短信登录发证服务

 \*/

authBySms() {

    let phoneNum = **this**.phoneNum; // 手机号

    let verifyCode ="12345"; // 获取到的手机验证码

    kztSdkModule.authBySms(phoneNum, verifyCode, res => {

        // {"error":"","code":"0"}

        console.log("authBySms" + JSON.stringify(res))

        modal.toast({

            message: res,

            duration: 1.5

        });

    });

}

##### 签发证书

/\*\*

 \* 证书签发

 \*/

certIssue() {

    // 需要先调用 sendAuthSms和 authBySms进行认证

    let pin = **this**.pin; // 证书PIN码

    kztSdkModule.certIssue(pin, res => {

        console.log("certIssue " + JSON.stringify(res))

        modal.toast({

            message: res,

            duration: 1.5

        });

    });

}

##### 获取本地证书列表

查看[本地证书信息](#_本地证书信息)

/\*\*

 \* 获取本地证书列表

 \*/

getLocalCertList() {

    kztSdkModule.getLocalCertList(res => {

    // {"code":"0","result":[{"subjectName":{"CN":"张三"},"subjectDN":"C=CN,L=西安市,OU=根机构,emailAddress=xxx@qq.com,CN=张三","actionDate":"2024-03-28 11:03:36","subjectId":"xxx","signCertSn":"xxx","signFlag":"1","devName":"smf","issuerName":{"CN":"local\_ca\_sm2"},"SN":"9xxx0","issuerDN":"C=CN,CN=local\_ca\_sm2","validDate":"2026-03-28 11:03:35","appName":"ZmMazuDBujaUVRL","containerName":"zgkhjtkyhwvZptv","certType":"SM2"}]}

        console.log("getLocalCertList " + JSON.stringify(res))

        modal.toast({

            message: res,

            duration: 1.5

        });

**if** (res) {

            let certList = res.result;

**if** (certList.length > 0) {

**this**.subjectId = certList[0]["subjectId"]

            }

        }

    });

}

##### 通过证书登录认证服务

/\*\*

 \* 登录认证服务

 \*/

login() {

    kztSdkModule.login({

        authnType: "pki\_cert", // 认证类型

        authnTypeInstanceId: "pki\_cert", // 认证子类型，和认证类型保持一致即可

        certInfo: {

            // 从getLocalCertList结果中获取subjectId

            subjectId: **this**.subjectId,

            // 用户输入证书密码

            pin: **this**.pin

        }

    }, res => {

        console.log("login " + JSON.stringify(res))

        modal.toast({

            message: res,

            duration: 1.5

        });

    });

}

##### 获取登录二维码信息

/\*\*

 \* 获取登录二维码信息

 \*/

getQrInfo() {

    let appId = "0"; // 应用ID

    let qrInfo = **this**.qrcodeInfo; // 扫描的二维码信息

    kztSdkModule.getQrInfo(appId, qrInfo, res => {

        console.log("getQrInfo " + JSON.stringify(res))

        modal.toast({

            message: res,

            duration: 1.5

        });

        let result = res.result;

**if**(result){

**this**.qrcodeRandom = result.random;

        }

    });

}

##### 授权二维码登录

/\*\*

 \* 授权二维码登录

 \*/

confirmQrInfo() {

    let appId = "0"; // 应用ID

    let qrInfo =  **this**.qrcodeInfo; // 扫描的二维码信息

    let random = **this**.qrcodeRandom // getQrInfo结果中的random

    kztSdkModule.confirmQrInfo(appId, qrInfo, random, res => {

        console.log("confirmQrInfo " + JSON.stringify(res))

        modal.toast({

            message: res,

            duration: 1.5

        });

    });

}

##### 修改证书PIN码

/\*\*

 \* 修改证书pin

 \*/

changePin() {

    let pin = **this**.pin // 证书PIN码

    let newPin = "654321" // 新的证书PIN码

    kztSdkModule.changePin(pin, newPin, res => {

        // {"error":"","code":"0"}

        console.log("changePin " + JSON.stringify(res))

        modal.toast({

            message: res,

            duration: 1.5

        });

    });

}

##### 重置证书PIN码

/\*\*

 \* 重置证书pin

 \*/

resetPin() {

    // 需要先调用 sendAuthSms和 authBySms进行认证

    let pin = **this**.pin; // 证书PIN码

    kztSdkModule.resetPin(pin, res => {

        // {"error":"","code":"0"}

        console.log("resetPin" + JSON.stringify(res))

        modal.toast({

            message: res,

            duration: 1.5

        });

    });

}

##### 更新/延期证书

/\*\*

 \* 证书更新/延期

 \*/

certDelay() {

    // 需要先调用 sendAuthSms和 authBySms进行认证

    let pin = **this**.pin; // 证书PIN码

    kztSdkModule.certDelay(pin, res => {

        // {"error":"","code":"0"}

        console.log("certDelay " + JSON.stringify(res))

        modal.toast({

            message: res,

            duration: 1.5

        });

    });

}

##### 废除证书

/\*\*

 \* 证书废除

 \*/

certRevoke() {

    // 需要先调用 sendAuthSms和 authBySms进行认证

    let pin = **this**.pin; // 证书PIN码

    kztSdkModule.certRevoke(pin, res => {

        // {"error":"","code":"0"}

        console.log("certRevoke " + JSON.stringify(res))

        modal.toast({

            message: res,

            duration: 1.5

        });

    });

}

##### 删除本地证书

/\*\*

 \* 删除本地证书

 \*/

deleteLocalCert() {

    kztSdkModule.deleteLocalCert(res => {

        // {"error":"","code":"0"}

        console.log("deleteLocalCert " + JSON.stringify(res))

        modal.toast({

            message: res,

            duration: 1.5

        });

    });

}

##### 登出

/\*\*

 \* 登出

 \*/

logout() {

    kztSdkModule.logout(res => {

        // {"error":"","code":"0"}

        console.log("logout " + JSON.stringify(res))

        modal.toast({

            message: res,

            duration: 1.5

        });

    });

}

### SDK定义说明

#### DeviceInfo

移动设备相关信息，用于身份认证登录策略初始化。

**public** **class** DeviceInfo {

**private** String appVersion;//APP版本

**private** String deviceId;//设备标识

**private** String deviceIp;//设备ip地址

**private** String deviceMac;//设备mac地址

**private** String deviceModel;//设备型号

**private** String deviceName;//设备名称

**private** String deviceOs;//设备系统

**private** String deviceOsVersion;//设备系统版本

}

#### TeamConfig

项目配置类，可以根据项目要求进行配置，实现界面或SDK内的不同处理。

**public** **class** TeamConfig {

**private** String appName; // 登录界面logo

**private** **int** appIcon; // 登录界面title

**private** **boolean** showConfigServer; // 是否显示服务配置

**private** String configServer; // 默认配置服务地址

**private** **boolean** allowImportCert; // 是否允许导入证书

**private** **boolean** allowApplyCert; // 证书登录是否允许申请证书

**private** **boolean** useOnlineMode; //使用在线发证模式，true为在线模式  false为离线模式

**private** **boolean** isThirdAuth; // 是否是第三方认证

**private** **boolean** defaultAutoLogin = **true**; // 默认自动登录状态

**private** CdsAuthType authType = CdsAuthType.PWD; // 发证服务认证方法

**private** **boolean** allowShowPwd; // 是否需要显示密码

**private** **int** httpTimeOut = 30; // 网络请求超时时间

**private** **int** httpRetryCount = 0; // 网络请求重试次数

}

#### ResultBean<T>

身份认证接口响应结构体

**public** **class** ResultBean<T> {

**private** String code;

**private** String error;

**private** String extUserAttrs;

**private** Object result; // 原始返回数据格式

**private** T result;

**private** Integer timestamp;

**private** String traceId;

**private** String userUnique;

}

#### KztCallback

**public** **interface** KztCallback {

**void** onSuccess(T result);

**void** onFail(String code, String error);

}

#### 本地证书信息

**public** **class** CertInfoBean {

**private** String devID;

**private** String devName; // 设备名称

**private** String manufacturer; // 设备厂商

**private** String appName; // 应用名称

**private** String containerName; // 容器名称

**private** String hasPIN; // 是否有 PIN 码

**private** String hasFinger; // 是否具有指纹，0：非指纹 key；1： 指纹 key，应用未录入指纹；2：指 纹 key，存在指纹；0xff：未知类 型

**private** String signFlag; // 证书用途, 1 表示签名证书, 0 表示 加密证书

**private** String subjectId; // 证书对应的用户id

**private** String subjectDN; // 使用者DN项

**private** SubjectNameBean subjectName; // 使用者，所有子项，无数据时返回{}

**private** String issuerDN; // 颁发者DN项

**private** IssuerNameBean issuerName; // 颁发者，所有子项，无数据时返回{}

**private** String SN; // 序列号

**private** String actionDate; // 生效时间

**private** String validDate; // 失效时间

**private** String certType; // 证书类型

**private** String keyUsage;

**private** String signCertSn;//签名证书序列号

}

#### 证书状态

**public** **enum** CertStatus {

    NOT\_FOUND(-99, "未定义"),

    CERT\_ST\_NOT\_REGISTERED(-2, "未注册"),

    CERT\_ST\_UNREGISTERED(-1, "已注销"),

    CERT\_ST\_NOT\_EXIST(0, "证书不存在"),

    CERT\_ST\_EXIST(1, "证书存在"),

    CERT\_ST\_REGISTER2REVIEW(1000, "证书申请待审核"),

    CERT\_ST\_ISSUE\_REFUSED(1001, "证书申请审核未通过"),

    CERT\_ST\_TOBE\_ISSUE(1002, "待发证"),

    CERT\_ST\_ISSUED(2000, "已发证"),

    CERT\_ST\_UPDATE2REVIEW(2001, "更新待审核"),

    CERT\_ST\_TOBE\_UPDATE(2002, "待更新"),

    CERT\_ST\_POSTPONE2REVIEW(2003, "延期待审核"),

    CERT\_ST\_TOBE\_POSTPONE(2004, "待延期"),

    CERT\_ST\_FREEZED(2005, "已冻结"),

    CERT\_ST\_CERT\_REVOKED(2006, "证书已废除"),

    CERT\_ST\_REAPPLY2REVIEW(2007, "重发待审核"),

    CERT\_ST\_TOBE\_RECOVER(2008, "待恢复"),

    CERT\_ST\_TOBE\_REISSUE(2009, "待重发"),

    CERT\_ST\_FREEZE2REVIEW(2010, "冻结待审核"),

    CERT\_ST\_TOBE\_FREEZE(2011, "待冻结"),

    CERT\_ST\_UNFREEZE2REVIEW(2012, "解冻待审核"),

    CERT\_ST\_TOBE\_UNFREEZE(2013, "待解冻"),

    CERT\_ST\_KEYUPDATE2REVIEW(2014, "密钥更新待审核"),

    CERT\_ST\_TOBE\_KEYUPDATE(2015, "待更新密钥"),

    CERT\_ST\_RECOVER2REVIEW(2016, "恢复待审核"),

    CERT\_ST\_REVOKED2REVIEW(2017, "废除待审核"),

    CERT\_ST\_REVOKED2REVIEW2(2018, "冻结废除待审核"),

    CERT\_ST\_TOBE\_REVOKED(2019, "待废除"),

    CERT\_ST\_FREEZED2REVOKED(2021, "冻结待废除"),

    CERT\_ST\_UNLOCK2REVIEW(2022, "解锁申请待审核"),

    CERT\_ST\_TOBE\_UNLOCKED(2023, "证书待解锁"),

    CERT\_ST\_LOCKED(2024, "证书锁定"),

    CERT\_ST\_EXPIRED(2025, "证书已过期"),

    CERT\_ST\_UNEFFECT(2026, "证书未生效"),

    CERT\_ST\_EXPIRED\_SOON(2027, "证书即将过期");

}

#### TicketInfoBean票据信息

**public** **class** TicketInfoBean {

**private** String ticket; // 票据

**private** String zone; // 节点信息，验证ticket时需同时携带

**private** **int** expireIn; // //票据过期时间，单位：s

**private** **long** expireTime; // 过期秒数时间戳

}

# 部政务用户统一身份认证身份信息同步接口

部统一身份认证管理系统集中管理了移动应急的用户身份并为各移动应用提供身份信息。

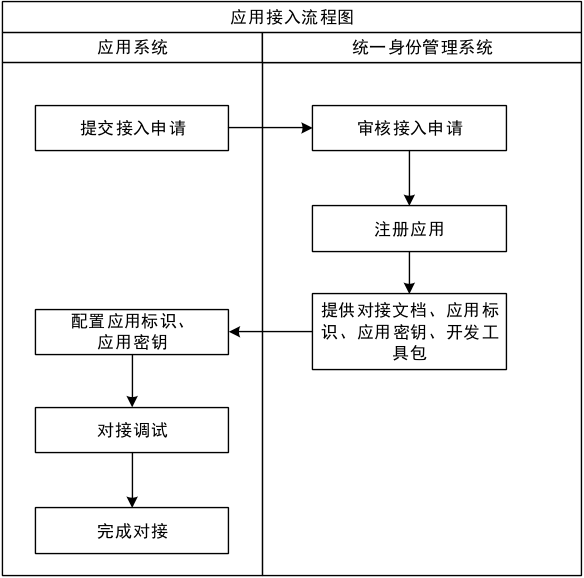
本章说明了部统一身份认证与移动应用系统对接接口。描述应用系统与统一身份管理系统进行身份信息同步对接的详细流程，并对对接开发过程中涉及的身份数据模型、数据同步接口规范、签名验签机制进行说明

## 应用系统集成

### 应用系统对接流程

应用系统集成统一身份管理系统身份信息同步接口，需先在测试环境进行集成对接，对接完成后，再在正式系统进行上线。

对接基本流程如下：

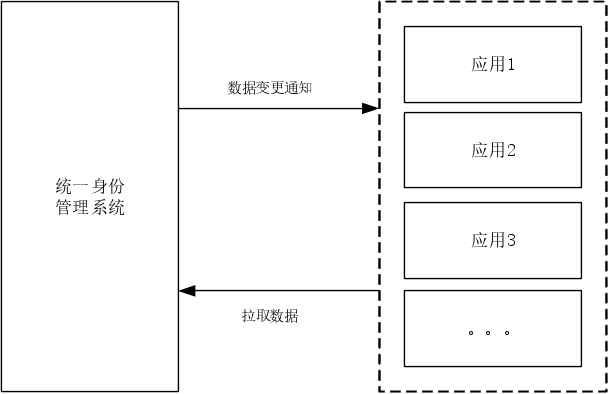


应用系统负责人提交应用信息，申请接入统一身份管理系统；

1. 统一身份管理系统负责人对应用系统负责人提交的信息进行审批，如果发现问题与应用系统负责人进行沟通；
2. 统一身份管理系统负责人在统一身份管理系统注册应用，生成应用身份信息，如应用系统标识appKey、应用系统密钥appPwd等；
3. 统一身份管理系统负责人将对接信息提供给应用系统负责人，如对接文档、应用系统标识、应用系统密钥、开发工具包；应用系统负责人需确保密钥安全，避免泄露；
4. 应用系统开发人员配置应用系统标识、应用系统密钥，根据对接文档进行对接开发、联调和测试。
5. 联调测试通过后，统一身份管理系统负责人确认对接结果，确认后按上述流程进行正式环境配置上线。

### 身份信息同步机制

统一身份管理系统集中管理组织及用户资源，提供信息变更订阅和信息查询服务接口。具体流程如下图所示。



当有资源更新时，统一身份管理系统发送数据变更通知给各个应用系统，应用系统接收到数据更新通知消息后（参见“[数据更新消息通知](#_数据更新消息通知)”），调用数据同步接口拉取数据（参见“[机构信息同步接口](#_机构信息同步接口)”、“[人员信息同步接口](#_人员信息同步接口)”）获取机构、人员数据。

统一身份管理系统数据同步接口包含同步序列号字段，该字段按数据变更顺序自增。首次同步时，通过将同步序列号值设为0拉取全量数据。同步时，需先同步机构数据，再同步人员数据，以确保依赖数据已经存在。应用系统需要保留每次数据同步完成时的同步序列号，用于下次拉取数据，以实现增量数据同步。

应用系统也可以根据自己的需求，通过定时触发或手动触发的方式，从统一身份管理系统同步数据。

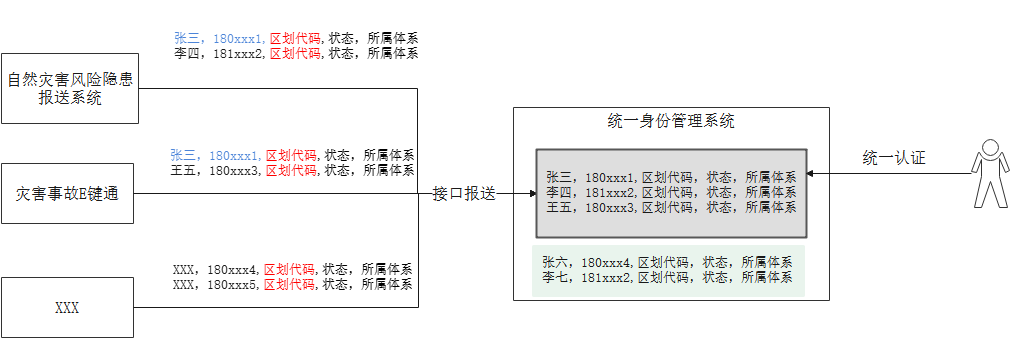
统一身份管理系统对每一个同步接口调用进行身份验证，应用系统调用接口时，需要根据指定签名规则对接口参数进行签名，服务器通过签名信息验证请求的发送者身份，参见“[接口签名规则](#_接口签名规则)”。

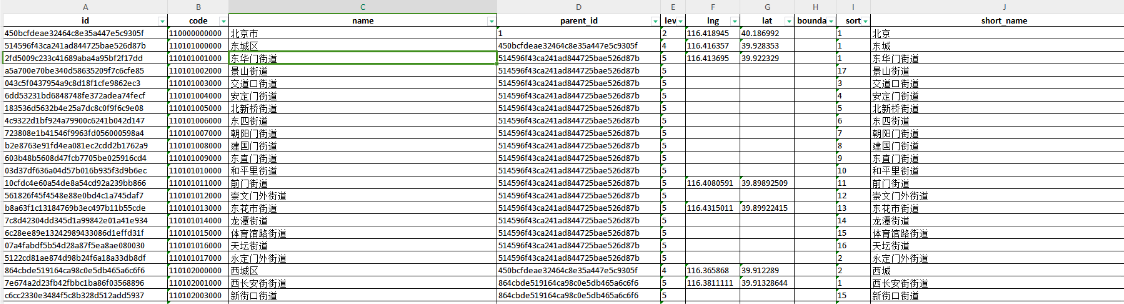
### 过渡方案

#### 数据上报

当前部分业务系统数据梳理工作量较大，为保证项目上线时间，考虑通过数据上报汇总的方式，先完成初步的数据收集及满足统一认证要求。

身份数据收集方案如下：



1. 各业务系统将自身需要参与统一认证的政务用户（包括部省市县及县级以下），通过“[人员上报接口（乡镇街道）](#_人员上报接口（乡镇街道）)”上报至统一身份管理系统。上报信息需包括姓名、手机号、区划代码、区划名称（机构名称）、状态、所属体系。所属体系按应急机构编码规范定义，具体参见“[所属体系（ORG\_SYSTEM）](#_所属体系（ORG_SYSTEM）)”。统一身份管理系统提供统一区划规范，参考如图： 
2. 统一身份管理系统以手机号做唯一标识，将上报数据保存在本地用于统一认证，同时记录该数据来源业务系统。 后续改记录更新只允许该业务系统操作。其他业务系统上报数据有手机号重复的人员时，系统提示用户已存在。手机号变更操作只可在统一身份管理系统进行。
3. 统一身份管理系统结合当前已有政务体系数据及业务系统上报数据，完成政务人员身份认证。

## 接口描述

### 数据更新消息通知

当统一身份管理系统中资源信息发生变更时，会通过HTTP协议发送通知消息给应用系统。应用系统接受到通知消息可调用对应的资源同步接口获取增量数据。应用系统需要实现并提供该接口，以保证数据同步的及时性。

|  |  |
| --- | --- |
| **URL** | 数据接收方提供URL地址，登记到统一身份管理系统中 |
| **通知方式** | POST |
| **通知参数** | {  "resType":"资源类型"  } |
| **请求头** | |  |  |  | | --- | --- | --- | | **参数** | **参数说明** | **是否必选** | | once | 随机数，32位，可用UUID表示 | 必选 | | ts | 时间戳 | 必选 | | appKey | 应用标识 | 必选 | | signMethod | 签名方法 | 必选，参见“[签名方法](#_签名方法)” | | signData | 参数的签名 | 必选 | |
| **参数说明** | resType为资源类型，可选值包括user、org，参见“[资源类型](#_资源类型)”。 |
| **示例** | {  "resType":"org"  } |
| **响应信息** | {  "resultCode":200, //与响应头一致的HTTP状态码  "resultMessage":"" //结果信息描述  } |
| **备注** | 统一身份系统不以响应信息具体内容区分通知发送结果。应用系统接收通知后，需返回Http状态码200。 |

### 机构信息同步接口

#### 机构信息字段说明

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段标识** | **字段描述** | **类型长度** | **是否可为查询条件** | **备注** |
| ORG\_ID | 机构ID | VARCHAR(32) | 是 | 机构在统一身份管理系统内唯一标识，全生命周期内保持不变。  应用系统获取机构标识后，需本地保留，以便跟踪机构的全生命周期变化 |
| ORG\_NAME | 机构名称 | VARCHAR(100) | 是 |  |
| ORG\_NAME\_SHORT | 机构简称 | VARCHAR(128) |  |  |
| ORG\_NO | 机构编码 | VARCHAR(64) | 是 | 根据应急部机构编码规范定义的机构业务标识。  机构全生命周期内存在变更可能。 |
| ORG\_TYPE | 是否正式部门 | INT | 是 | 参见机构数据字典“[机构类型（ORG\_TYPE）](#_机构类型（ORG_TYPE）)”  辅助机构分为直属单位、内设机构，直属单位的机构编码为其上级机构编码+“Z”；内设机构的机构编码与其上级机构相同。辅助机构只作为组织机构层级展示用，不关联用户。 |
| ADDRESS | 地址 | VARCHAR(100) |  |  |
| PARENT\_ID | 上级机构ID | VARCHAR(32) | 是 | 根机构时，该字段为空。 |
| PARENT\_ORG\_NO | 上级机构编码 | VARCHAR(64) | 是 | 根机构时，该字段为空。 |
| SHOW\_ORDER | 显示顺序 | VARCHAR(64) |  | 排序号，默认为空，不为空时，为整形数字。建议按排序号、机构编码双字段排序。 |
| ORG\_STATUS | 状态 | INT | 是 | 参见机构数据字典“[机构状态（ORG\_STATUS）](#_机构状态（STATUS）)” |
| REMARK | 备注 | VARCHAR(500) |  |  |
| UPDATE\_TIME | 更新时间 | LONG |  | 时间的长整型表示 |

#### 机构信息查询接口

|  |  |
| --- | --- |
| **URL** | http://ip:port/id-sync/v5/urm/service/V3/res/orgs?[syncSequence=xxxxx][&ORG\_NO=xxxxx][&ORG\_NAME=xxxxx][&PARENT\_ID=xxxxX] |
| **调用方式** | GET |
| **参数说明** | **syncSequence**: 同步序列号，必填；为0时，表示查询所有数据；不为0时，查询指定序号后的所有数据；  **ORG\_NO**：机构编号，可选；  **ORG\_NAME**：机构名称，可选；  **PARENT\_ID**：父机构ID，可选；  具体支持的查询参数字段参见机构信息字段说明中“是否可为查询条件”标识。 |
| **请求示例** | http://127.0.0.1/service/V5/res/orgs?syncSequence=1584868514781589503 |
| **请求头** | descript |
| **返回值** | {  "resultCode": 200,  "resultMsg": "正常",  "data": [  {  "SYNC\_SEQUENCE": "426523467044438017",  "ORG\_ID": "423314327558156288",  "ORG\_NAME": "9d22fdd10225719b7e72c491593b5cb2",  "ORG\_NAME\_SHORT": "测试",  "ORG\_NO": "02020100000000000",  "SHOW\_ORDER": "1",  "PARENT\_ORG\_NO": "02020000000000000",  "UPDATE\_TIME": 1702380627000,  "PARENT\_ID": "421166296043429888",  "ORG\_TYPE": "10",  "ORG\_STATUS": 2000  }  ],  "total":1,  "dataSize":1,  "nextSyncSequence": "426523467044438017"  } |
| **返回值说明** | **resultCode**：查询结果状态编码，200为成功，参见“[请求结果状态编码说明](#_请求结果编码说明)”；  **resultMsg**：查询结果说明；  **data**: 数据明细；  **total**:符合条件的数据总数；无符合条件数据时返回0；  **dataSize**:当前返回的数据记录数；  **nextSyncSequence**：数据同步序列号；如果本次查询返回的nextSyncSequence与请求参数中的一致，且total为0，表明数据已查询完毕，应用方需保存当前nextSyncSequence数据，用于下次查询； |
| **备注** | 同步时的数据按同步序列号由小到大返回，返回的数据中无法保证上下级机构的返回顺序。应用系统同步机构数据时，需注意机构数据的上下级关系，优先完成上级机构数据的处理，确保机构层级的完整性。应用系统需确认本地或者当前批次中是否存在上级数据，如果不存在，可以根据PARENT\_ID获取上级机构数据。 |

### 人员信息同步接口

#### 人员信息字段说明

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **字段标识** | **名称** | **类型长度** | **是否可为查询条件** | **备注** |
| USER\_ID | 人员ID | VARCHAR(32) | 是 | 人员在统一身份管理系统内唯一标识，全生命周期内保持不变。 |
| IDCARD\_UUID | 身份UUID（身份证号对应的） | VARCHAR(64) | 是 | 人员全局唯一标识，全生命周期内保持不变。  应用系统获取人员标识后，需本地保留，以便跟踪人员的全生命周期变化 |
| USER\_NAME | 姓名 | VARCHAR(256) | 是 |  |
| USER\_NAME\_PINYIN | 姓名拼音 | VARCHAR(256) |  |  |
| JOB\_TITLE | 职称 | VARCHAR(512) | 是 | 参见政务人员字典“[职称（JOB\_TITLE）](#_职称（JOB_TITLE）)” |
| MOBILE | 手机号 | VARCHARE(32) | 是 |  |
| EMAIL | 邮箱 | VARCHAR(128) |  |  |
| SEX | 性别 | INT |  | 参见政务人员字典“[性别（SEX）](#_性别（SEX）)” |
| REMARK | 备注 | VARCHAR(4000) |  |  |
| UPDATE\_TIME | 更新时间 | LONG |  | 时间的长整型表示 |
| USER\_JOB\_ID | 人员岗位ID | VARCHAR(32) | 是 | 岗位记录的标识 |
| ORG\_ID | 所属机构 | VARCHAR(32) | 是 | 任职机构ID，业务系统需保留任职机构ID，以便同步任职状态变更 |
| ORG\_NO | 机构编码 | VARCHAR(64) | 是 | 任职机构编码。机构编码存在变更可能，业务系统应以任职机构ID标识任职信息。 |
| JOB\_TYPE | 用户类型 | INT | 是 | 任职类型，参见政务人员字典“[任职类型（JOB\_TYPE）](#_任职类型（JOB_TYPE）)” |
| JOB\_POSITION | 职务 | VARCHAR(300) |  | 任职职务，参见政务人员字典“[职务（JOB\_POSITION）](#_职务（JOB_POSITION）)” |
| VALID\_TIME | 任职有效期 | LONG |  | 非正式任职时提供该信息 |
| OFFICE\_PHONE | 办公电话 | VARCHAR(64) |  |  |
| OFFICE\_ADDRESS | 办公地址 | VARCHAR(128) |  |  |
| SHOW\_ORDER | 显示顺序 | VARCHAR(64) |  | 排序号，默认为空，不为空时，为整形数字 |
| JOB\_STATUS | 任职状态 | INT |  | 任职状态，参见政务人员字典“[任职状态（JOB\_STATUS）](#_任职状态（JOB_STATUS）)”。默认为有效。 |

#### 人员信息查询接口

|  |  |
| --- | --- |
| **URL** | http://ip:port/id-sync/v5/urm/service/V3/res/users?[syncSequence=xxxxx][&USER\_NAME=xxxxx][&IDCARD\_UUID=xxxxx] |
| **调用方式** | GET |
| **参数** | **syncSequence:** 同步序列号，必填；为0时，表示查询所有数据；不为0时，查询指定序号后的所有数据；  **USER\_NAME**：姓名，可选；  **IDCARD\_UUID**：身份标识，可选；  具体支持的查询参数字段参见人员信息字段说明中“是否可为查询条件”标识。 |
| **请求示例** | http://127.0.0.1/service/V5/res/users?syncSequence=1584868514781589504 |
| **请求头** | descript |
| **返回值** | {  "resultCode": 200,  "resultMsg": "正常",  "data": [  {  "SYNC\_SEQUENCE": "426523591602683905",  "USER\_ID": "409477449139945472",  "IDCARD\_UUID": "409477449089613824",  "USER\_NAME": "张洋",  "MOBILE": "13333333333",  "SEX": 1,  "UPDATE\_TIME": 1702444772000,  "USER\_JOB\_ID": "409477449517432833",  "ORG\_ID": "409472671261458432",  "ORG\_NO": "01000000000000000",  "JOB\_TYPE": 10,  "JOB\_POSITION": "70",  "JOB\_STATUS": 2000,  "VALID\_TIME": 1703692800000,  "SHOW\_ORDER": "3",  "OFFICE\_ADDRESS": "2222",  "OFFICE\_PHONE": "1111"  }  ],  "total":1,  "dataSize":1,  "nextSyncSequence": "426523591602683905"  } |
| **返回值说明** | **resultCode**：查询结果状态编码，200为成功，参见“[请求结果状态编码说明](#_请求结果编码说明)”；  **resultMsg**：查询结果说明；  **data**: 数据明细；  **total:**符合条件的数据总数；无符合条件数据时返回0；  **dataSize:**当前返回的数据记录数；  **nextSyncSequence：**数据同步序列号；如果本次查询返回的nextSyncSequence与请求参数中的一致，且total为0，表明数据已查询完毕，应用方需保存当前nextSyncSequence数据，用于下次查询； |
| **备注** | 用户数据以任职信息为粒度返回，每条记录对应一条用户任职信息。如果用户在多个机构任职，则存在多条记录。任职记录注销时，则任职状态为注销。 |

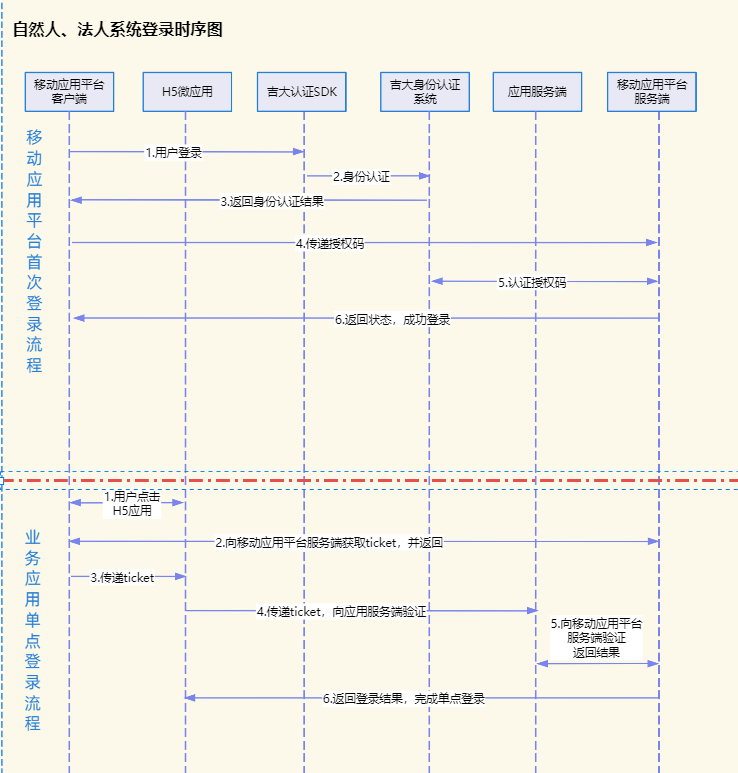
#### 人员上报接口（乡镇街道）

|  |  |
| --- | --- |
| **URL** | http://ip:port/id-sync/v5/townusers/upload |
| **调用方式** | POST |
| **请求参数** | [ {     "USER\_NAME": "张三", //姓名     "MOBILE": "18011021314", //手机号码  "AREA\_CODE": "110101003000", //区划代码  "AREA\_NAME": "交道口街道", //区划名称（机构名称）  "ORG\_SYSTEM": "01", //所属体系,参见字典“[所属体系（ORG\_SYSTEM）](#_所属体系（ORG_SYSTEM）)”  "STATUS": 1// 1有效，0无效 } ] |
| **请求示例** | http://ip:port/id-sync/v5/townusers/upload |
| **请求头** | descript |
| **返回值** | { "resultCode": 200, "resultMsg": "正常"  } |
| **返回值说明** | **resultCode**：查询结果状态编码，200为成功，参见“[请求结果状态编码说明](#_请求结果编码说明)”；  **resultMsg**：查询结果说明； |
| **备注** | 1. 同一用户只允许创建该用户的业务系统进行变更。其他业务系统不可进行更新。其他业务系统可根据需要，单独查询用户进行本地更新。 |

# 部自然人、法人统一身份认证

部自然人、法人用户统一身份认证系统用于实现移动应急业务的自然人和法人的身份认证与单点登录功能。

## 自然人、法人登录时序图



1. **自然人、法人登录时序图**

## 新增自然人信息

### 接口描述

此接口实现新增自然人信息

|  |  |
| --- | --- |
| 接口类型： | Restful |
| 接口地址： | authcenter/userInfo/savePersonInfo.do |
| 功能描述： | 新增自然人信息 |

### 接口入参

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 参数名 | 参数类型 | 是否必填 | 参数说明 |
| 1 | name | String | 是 | 姓名 |
| 2 | idCard | String | 是 | 身份证号码 |
| 3 | certEffDate | String | 是 | 证件有效日期 日期格式：YYYYMMDD |
| 4 | certExpDate | String | 是 | 证件失效日期 日期格式：YYYYMMDD |
| 5 | mobilePhone | String | 是 | 手机号码 |
| 6 | password | String | 否 | 密码（dataType为1时，必填） |
| 7 | smsCode | String | 否 | 短信验证码（dataType为1时，必填） |
| 8 | dataType | String | 是 | 1.注册；2.历史人员推送 |
| 9 | dataFrom | String | 是 | 数据来源 |
| 请求及响应参数示例：  Request：  {  "name":"xx",  "idCard":"xxx",  "mobilePhone":"18688886666",  "certEffDate":"20211116",  "certExpDate":"20311116",  "password":"665143bed092ba0040e522493f0b21e20bb2018909626a18be7bf948f0ccbd7b8d80c966b69a6d1f7b10686c9ff3df67",  "smsCode":"123456",  "dataType":"1",  "dataFrom":"app"    }  Response：  {"flag":true,"message":"成功"} | | | | |

### 接口返回值

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 参数名 | 参数类型 | 参数说明 |
| 1 | flag | Boolean | 是否成功 true：成功 false：失败 |
| 2 | message | String | 响应信息 |

## 新增法人信息

### 接口描述

此接口实现新增法人信息功能

|  |  |
| --- | --- |
| 接口类型： | Restful |
| 接口地址： | authcenter/userInfo/saveEntInfo.do |
| 功能描述： | 新增法人信息 |

### 接口入参

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 参数名 | 参数类型 | 是否必填 | 参数说明 |
| 1 | entType | String | 是 | 1:企业、农专社,2:事业单位,3:社团、民非、基金会,4:政府机关,:5个体 |
| 2 | entName | String | 是 | 企业名称 |
| 3 | entUnitCode | String | 是 | 统一社会信用代码 |
| 4 | legalName | String | 是 | 法定代表人姓名 |
| 5 | legalCard | String | 是 | 法定代表人身份证号码 |
| 6 | agentName | String | 是 | 经办人姓名 |
| 7 | mobilePhone | String | 是 | 经办人手机号码 |
| 8 | password | String | 否 | 密码 |
| 9 | smsCode | String | 否 | 短信验证码 |
| 10 | dataType | String | 是 | 1.注册；2.历史人员推送 |
| 11 | dataFrom | String | 是 | 数据来源 |
| 请求及响应参数示例：  Request：  {  "entType":"2",  "entName":"xx单位",  "entUnitCode":"xxx",  "legalName":"xxx",  "legalCard":"xxx",  "agentName":"18813881688",  "mobilePhone":"18813881688",  "password":"665143bed092ba0040e522493f0b21e20bb2018909626a18be7bf948f0ccbd7b8d80c966b69a6d1f7b10686c9ff3df67",  "smsCode":"123456",  "dataType":"1",  "dataFrom":"app"    }  Response：  {"flag":true,"message":"成功"} | | | | |

### 接口返回值

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 参数名 | 参数类型 | 参数说明 |
| 1 | flag | Boolean | 是否成功 true：成功 false：失败 |
| 2 | message | String | 响应信息 |

## 登录认证

### 接口描述

此接口实现登录认证功能

|  |  |
| --- | --- |
| 接口类型： | Restful |
| 接口地址： | authcenter/userInfo/appLogin.do |
| 功能描述： | 登录认证 |

### 接口入参

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 参数名 | 参数类型 | 是否必填 | 参数说明 |
| 1 | appCode | String | 是 | 应用代码 |
| 2 | loginType | String | 是 | 登录方式（1.账号+密码；2.手机号+短信验证码） |
| 3 | userType | String | 是 | 人员类型（1.自然人；2.企业） |
| 4 | userName | String | 否 | 账号，企业统一社会信用代码/自然人证件号码（登录方式为账号+密码必填） |
| 5 | password | String | 否 | 密码（登录方式为账号+密码必填） |
| 6 | mobilePhone | String | 否 | 手机号码（登录方式为手机号+短信验证码必填） |
| 7 | smsCode | String | 否 | 短信验证码（登录方式为手机号+短信验证码必填） |
| 请求及响应参数示例：  Request：  {  "appCode":"app",  "loginType":"1,  "userType":"1,  "userName":"222xxx6914",  "password":"665143bed092ba0040e522493f0b21e20bb2018909626a18be7bf948f0ccbd7b8d80c966b69a6d1f7b10686c9ff3df67",  "mobilePhone":"",  "smsCode":""  }  Response：  {"flag":true,"message":"成功","accessToken":"zxs123dasd"} | | | | |

### 接口返回值

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 参数名 | 参数类型 | 参数说明 |
| 1 | flag | Boolean | 是否成功 true：成功 false：失败 |
| 2 | message | String | 响应信息 |
| 3 | accessToken | String | token |

## 获取短信验证码

### 接口描述

此接口实现获取短信验证码功能

|  |  |
| --- | --- |
| 接口类型： | Restful |
| 接口地址： | authcenter/userInfo/appSendSMS.do |
| 功能描述： | 获取短信验证码 |

### 接口入参

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 参数名 | 参数类型 | 是否必填 | 参数说明 |
| 1 | appCode | String | 是 | 应用代码 |
| 2 | mobilePhone | String | 是 | 手机号码 |
| 请求及响应参数示例：  Request：  {  "appCode":"app",  "mobilePhone":"18813881688"  }  Response：  {"flag":true,"message":"成功","smsCode":"123123"} | | | | |

### 接口返回值

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 参数名 | 参数类型 | 参数说明 |
| 1 | flag | Boolean | 是否成功 true：成功 false：失败 |
| 2 | message | String | 响应信息 |
| 3 | smsCode | String | 短信验证码 |

## Token验证

### 接口描述

此接口实现token验证功能

|  |  |
| --- | --- |
| 接口类型： | Restful |
| 接口地址： | authcenter/userInfo/appTokenVerify.do |
| 功能描述： | token验证 |

### 2.2接口入参

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 参数名 | 参数类型 | 是否必填 | 参数说明 |
| 1 | appCode | String | 是 | 应用代码 |
| 2 | accessToken | String | 是 | token |
| 请求及响应参数示例：  Request：  {  "appCode":"app",  "accessToken":"zxs123dasd"  }  Response：  {"flag":true,"message":"成功"} | | | | |

### 接口返回值

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 参数名 | 参数类型 | 参数说明 |
| 1 | flag | Boolean | 是否成功 true：成功 false：失败 |
| 2 | message | String | 响应信息 |

## 获取用户信息

### 接口描述

此接口实现获取用户信息功能

|  |  |
| --- | --- |
| 接口类型： | Restful |
| 接口地址： | authcenter/userInfo/appGetUserInfo.do |
| 功能描述： | 获取用户信息 |

### 接口入参

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 序号 | 参数名 | 参数类型 | 是否必填 | 参数说明 |
| 1 | appCode | String | 是 | 应用代码 |
| 2 | accessToken | String | 是 | token |
| 请求及响应参数示例：  Request：  {  "appCode":"app",  "accessToken":"zxs123dasd"  }  Response：  {  "flag":true,  "message":"成功",  "name":"张三",  "...":"..."  } | | | | |

### 接口返回值

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 参数名 | 参数类型 | 参数说明 |
| 1 | flag | Boolean | 是否成功 true：成功 false：失败 |
| 2 | message | String | 响应信息 |
| 3 | userType | String | 人员类型（1.自然人；2.企业） |
| 4 | name | String | 姓名  （userType=1 时返回此属性） |
| 5 | idCard | String | 身份证号码  （userType=1 时返回此属性） |
| 6 | certEffDate | String | 证件有效日期 日期格式：YYYYMMDD  （userType=1 时返回此属性） |
| 7 | certExpDate | String | 证件失效日期 日期格式：YYYYMMDD  （userType=1 时返回此属性） |
| 8 | mobilePhone | String | 手机号码  （userType=1 时返回此属性） |
| 9 | entType | String | 1:企业法人,2:机关事业单位法人,3:社团法人,5:个体工商户  （userType=2 时返回此属性） |
| 10 | entName | String | 企业名称  （userType=2 时返回此属性） |
| 11 | entUnitCode | String | 统一社会信用代码  （userType=2 时返回此属性） |
| 12 | legalName | String | 法定代表人姓名  （userType=2 时返回此属性） |
| 13 | legalCard | String | 法定代表人身份证号码  （userType=2 时返回此属性） |
| 14 | agentName | String | 经办人姓名  （userType=2 时返回此属性） |
| 15 | mobilePhone | String | 经办人手机号码  （userType=2 时返回此属性） |
| 16 | dataFrom | String | 数据来源 |

# 附录

## H5改造-聊天消息结构附录

### 文本消息

{

"chatId" : "chat-app-id" ,

"content": "张总，好久不见 "

"ctype" : 0

}

### 图片消息

### 文件消息

### 位置消息

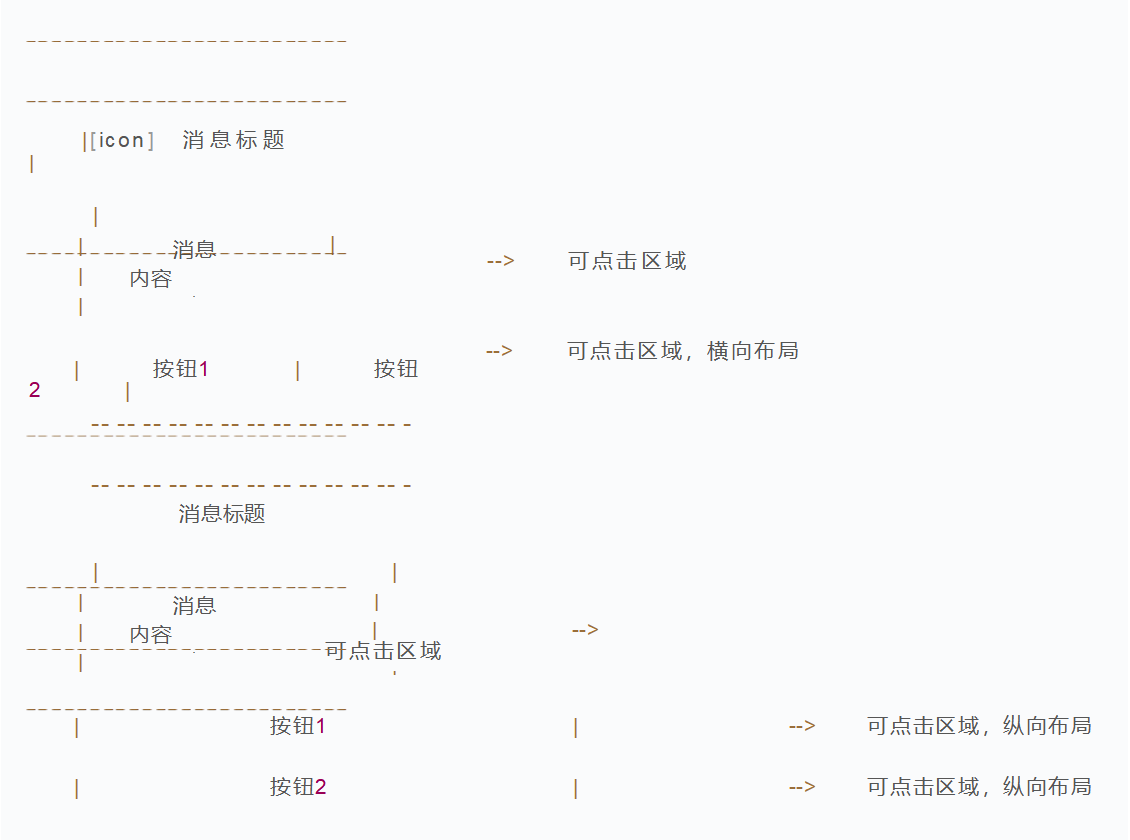
### 语音消息

### 链接消息

### 任务卡消息

任务卡消息可用于指定：当用户点击消息的特定区域时，跳转并展示指定的链接。可用于在消息发起方和APP之间进行协作。

任务卡消息的展示样式示例：



### 消息格式：

### 按钮控制

按钮可以设定最多可点击次数，在max字段中指定，用户点击按钮后，将在移动应急APP内加载对应的链接地址，如果加载成功，记为一次点击。达到最大可点击次数后，按钮将不可点击，如果设定了disable\_title字段，按钮标题内容将更改为指定的内容。

展示样式

任务卡内容支持使用HTML标签进行样式控制，建议使用标签如下：

 示意图如下：



### 短视频消息

### 文章

### 文章列表

## 部政务人员身份认证附录

### 政务人员属性信息

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **序号** | **字段标识** | **名称** | **类型长度** | **是否返回该字段** | **说明** |
| 1 | USER\_ID | 人员ID | CHAR(32) | 默认不返回； | 人员记录ID，更换所属机构后，该值会变。 |
| 2 | IDCARD\_UUID | 身份UUID | VARCHAR(64) | 默认返回 | 和身份证号对应，同一个身份证号第一次添加时系统自动生成，写到证书里。 |
| 3 | USER\_NAME | 姓名 | VARCHAR(256) | 默认返回 |  |
| 4 | IDCARD\_NO | 身份证号 | VARCHAR(64) | 默认不返回 | 敏感信息，默认不返回。 |
| 5 | JOB\_TITLE | 职称 | VARCHAR(512) | 默认不返回； | 职称字典表 |
| 6 | MOBILE | 手机号 | VARCHARE(32) | 默认不返回； | 敏感信息，默认不返回。OA当前需要该字段，所以返回。 |
| 7 | EMAIL | 邮箱 | VARCHAR(128) | 默认不返回； |  |
| 8 | ORG\_ID | 所属机构 | INT | 默认不返回； |  |
| 9 | TYPE | 用户类型 | INT | 默认不返回； | 类型字典表 |
| 10 | JOB\_POSITION | 职务 | VARCHAR(300) | 默认不返回； | 职务字典表 |
| 11 | ORG\_NAME | 机构名称 | VARCHAR(100) | 默认不返回； |  |
| 12 | ORG\_FULL\_NAME | 机构全称 | VARCHAR(300) | 默认不返回； |  |
| 14 | USER\_ACCOUNT | 账号 |  | 默认不返回； | 新增 |
| 15 | LEGAL\_PERSON\_ORG\_CODE | 机构编码 |  |  |  |
| 16 | DISTRICT\_CODE | 行政区划 |  |  |  |

### 对接完成测试流程

|  |  |  |
| --- | --- | --- |
| **序号** | **测试用例** | **预期结果** |
|  | 登录跳转，在未登录的条件下，直接访问应用系统 | 跳转到单点登录界面 |
|  | 登录跳转，在已登录的条件下，直接访问应用系统 | 直接进入应用系统 |
|  | 正常登录，输入认证系统地址，跳转单点登录页面，输入用户名/密码后，进入认证系统门户页面，点击应用系统图标，进入系统。 | 正常登录 |
|  | 正常登出，在认证系统登出 | 访问应用系统要重新登录 |
|  | 正常登出，在业务系统登出 | 单点登录系统同步登出 |

### 接口签名规则

针对没有数字证书和签名服务器的现场环境，统一身份管理系统会对每一个访问的请求进行身份验证，请求中需要包含appKey和由appPwd签名的signData。服务器通过签名信息验证请求的发送者身份。appKey、appPwd由统一身份管理系统在应用登记时生成，并分发给应用。其中appKey是用于标识应用身份的应用标识；appPwd是用于应用端签名和服务器端验证签名的应用密钥。appPwd必须严格保密，只有统一身份管理系统与对接应用知道。

签名算法支持国际算法和国密算法，建议使用国密算法。以下样例中以SHA-256说明。

#### 签名步骤

1. 使用请求参数构造规范化的请求字符串，请求参数主要由URL中，header中和body中参数组成。
2. 按照参数名称的ASCII顺序对请求中所有的请求参数（包括各请求URL后的自定义参数、请求header中的公共参数以及请求body中的JSON字符串参数，但不能包括请求头中的公共请求参数中signData参数本身）进行排序。

**说明：**

当使用GET方法提交请求时，这些参数就是请求URI中的参数部分（即URI中“?”之后由“&”连接的部分）。

根据ID查询或维护的接口中ID是作为路径存在的，所以需要将“ID”与ID的值组建为一个键值对进行签名，如：ID=aa03a5c692cf480b87887e0ff8cfe566。

其中POST/PUT的body参数一律以json字符串放在请求body中，body中存放JSON字符串将“bodyData”与JSON串组建为一个键值对进行签名，如：bodyData={"key1":"value1"}。若请求body中无参数，则无需加入bodyData键值对进行参数签名；

1. 对每个请求参数的名称和值进行编码。名称和值要使用UTF-8字符集进行URL编码，URL编码的编码规则是：

* 对于字符 A-Z、a-z、0-9以及字符“-”、“\_”、“.”、“~”不编码；
* 对于其他字符编码成“%XY”的格式，其中XY是字符对应ASCII码的16进制表示。比如英文的双引号（"）对应的编码就是%22
* 需要说明的是英文空格（ ）要被编码是%20，而不是加号（+）。

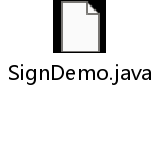
**说明：**

一般支持URL编码的库（比如Java中的java.net.URLEncoder）都是按照“application/x-wwwform-urlencoded”的MIME类型的规则进行编码的。实现时可以直接使用这类方式进行编码，把编码后的字符串中加号（+）替换成%20、星号（\*）替换成%2A、%7E替换回波浪号（~），即可得到上述规则描述的编码字符串。

1. 对编码后的参数名称和值使用英文等号（=）进行连接。
2. 再把英文等号连接得到的字符串按参数名称的字典顺序依次使用&符号连接，即得到规范化请求字符串。

对规范化请求字符串使用应用密钥进行签名，将得到的签名值作为signData参数添加到请求header中，即完成对请求签名的过程。

#### 签名示例



## 部政务人员身份认证信息同步附录

### 请求结果状态编码说明

与响应头一致的HTTP错误码。

|  |  |  |
| --- | --- | --- |
| **序号** | **状态值** | **说明** |
| 1 | 200 | 成功 |
| 2 | 20000 | 成功，与200兼容 |
| 3 | 40000 | 错误的客户端请求，包括缺少参数、参数格式错误 |
| 4 | 40001 | 没有通过身份认证，包括应用标识错误、应用密钥不正确、验证签名失败 |
| 5 | 40003 | 客户端通过了身份验证，但是不具有访问资源所需的权限 |
| 6 | 40004 | 请求资源不存在，包括请求url不正确 |
| 7 | 40005 | 客户端请求已经通过身份验证，但是所用的 HTTP 方法不在他的权限之内 |
| 8 | 40029 | 客户端请求次数超过限额，包括请求次数过多、请求被重放 |
| 9 | 50000 | 系统内部异常 |
| 10 | 60000 | 系统业务异常 |

### 接口相关枚举值

#### 签名方法

|  |  |  |
| --- | --- | --- |
| **序号** | **枚举值** | **枚举说明** |
| 1 | SM3 | 国产签名算法，建议用该算法。系统默认签名算法 |
| 2 | SHA-256 |  |

#### 资源类型

|  |  |  |
| --- | --- | --- |
| **序号** | **枚举值** | **枚举说明** |
| 1 | org | 机构资源 |
| 2 | user | 用户资源 |

### 接口相关字典

#### 机构字典

##### 机构类型（ORG\_TYPE）

| **字典值** | **字典描述** |
| --- | --- |
| -1 | 辅助 |
| 10 | 正式 |
| 20 | 临时 |

##### 机构状态（ORG\_STATUS）

| **字典值** | **字典描述** |
| --- | --- |
| 2000 | 正常状态 |
| -1000 | 注销状态 |

#### 人员字典

##### 性别（SEX）

| **字典值** | **字典描述** |
| --- | --- |
| 1 | 男 |
| 2 | 女 |

##### 职称（JOB\_TITLE）

| **字典值** | **字典描述** |
| --- | --- |
| 10 | 初级职称 |
| 20 | 中级职称 |
| 30 | 高级职称 |
| 40 | 无职称 |

##### 人员状态（STATUS）

| **字典值** | **字典描述** |
| --- | --- |
| 2000 | 正常状态 |
| -1000 | 注销状态 |

##### 职务（JOB\_POSITION）

| **字典值** | **字典描述** |
| --- | --- |
| 10 | 部党组书记 |
| 20 | 部长 |
| 30 | 副部长 |
| 40 | 政治部主任 |
| 50 | 部纪检组组长 |
| 60 | 总工程师 |
| 70 | 局长 |
| 80 | 副局长 |
| 90 | 司长 |
| 100 | 副司长 |
| 110 | 厅长 |
| 120 | 副厅长 |
| 128 | 纪检组组长 |
| 129 | 纪检组副组长 |
| 130 | 监察专员 |
| 140 | 一级巡视员 |
| 150 | 二级巡视员 |
| 160 | 主任 |
| 170 | 副主任 |
| 180 | 处长 |
| 190 | 副处长 |
| 193 | 正处级 |
| 194 | 副处级 |
| 196 | 书记 |
| 197 | 副书记 |
| 200 | 一级调研员 |
| 210 | 二级调研员 |
| 220 | 三级调研员 |
| 230 | 四级调研员 |
| 240 | 一级主任科员 |
| 250 | 二级主任科员 |
| 260 | 三级主任科员 |
| 270 | 四级主任科员 |
| 280 | 一级科员 |
| 290 | 二级科员 |
| 300 | 一级指挥长 |
| 310 | 二级指挥长 |
| 320 | 三级指挥长 |
| 330 | 四级指挥长 |

##### 任职类型（JOB\_TYPE）

| **字典值** | **字典描述** |
| --- | --- |
| 10 | 借调 |
| 20 | 挂职 |
| 30 | 正式 |

##### 任职状态（JOB\_STATUS）

| **字典值** | **字典描述** |
| --- | --- |
| 2000 | 正常状态 |
| -1000 | 注销状态 |

##### 所属体系（ORG\_SYSTEM）

| 字典值 | 字典描述 |
| --- | --- |
| 01 | 应急管理部门 |
| 02 | 消防救援机构 |
| 03 | 森林消防机构 |
| 04 | 地震机构 |
| 05 | 矿山机构 |
| SW | 商务部 |
| SL | 水利部 |
| ST | 生态环境部 |
| GB | 国务院办公厅 |
| ZR | 自然资源部 |
| GA | 公安部 |
| RS | 人力资源和社会保障部 |
| GS | 国家税务总局 |
| JT | 交通运输部 |
| RH | 中国人民银行 |
| TJ | 国家统计局 |
| QX | 中国气象局 |
| TL | 国家铁路局 |
| SG | 国家市场监督管理总局 |
| HG | 海关总署 |
| FG | 国家发展和改革委员会 |
| ZJ | 中国证券监督管理委员会 |
| MZ | 民政部 |
| CJ | 住房和城乡建设部 |
| ZG | 最高人民法院 |
| YZ | 中国邮政局 |
| LC | 国家林业和草原局 |

### 接口签名规则

针对没有数字证书和签名服务器的现场环境，统一身份管理系统会对每一个访问的请求进行身份验证，请求中需要包含appKey和由appPwd签名的signData。服务器通过签名信息验证请求的发送者身份。appKey、appPwd由统一身份管理系统在应用登记时生成，并分发给应用。其中appKey是用于标识应用身份的应用标识；appPwd是用于应用端签名和服务器端验证签名的应用密钥。**appPwd必须严格保密，只有统一身份管理系统与对接应用知道。**

**签名算法支持国际算法和国密算法，建议使用国密算法。以下样例中以SHA-256说明。**

#### 签名步骤

1. 使用请求参数构造规范化的请求字符串，请求参数主要由URL中，header中和body中参数组成。
2. 按照参数名称的ASCII顺序对请求中所有的请求参数（包括各请求URL后的自定义参数、请求header中的公共参数以及请求body中的JSON字符串参数，但不能包括请求头中的公共请求参数中signData参数本身）进行排序。

**说明：**

当使用GET方法提交请求时，这些参数就是请求URI中的参数部分（即URI中“?”之后由“&”连接的部分）。

根据ID查询或维护的接口中ID是作为路径存在的，所以需要将“ID”与ID的值组建为一个键值对进行签名，如：ID=aa03a5c692cf480b87887e0ff8cfe566。

其中POST/PUT的body参数一律以json字符串放在请求body中，body中存放JSON字符串将“bodyData”与JSON串组建为一个键值对进行签名，如：bodyData={"key1":"value1"}。若请求body中无参数，则无需加入bodyData键值对进行参数签名；

1. 对每个请求参数的名称和值进行编码。名称和值要使用UTF-8字符集进行URL编码，URL编码的编码规则是：

* 对于字符 A-Z、a-z、0-9以及字符“-”、“\_”、“.”、“~”不编码；
* 对于其他字符编码成“%XY”的格式，其中XY是字符对应ASCII码的16进制表示。比如英文的双引号（"）对应的编码就是%22
* 需要说明的是英文空格（ ）要被编码是%20，而不是加号（+）。

**说明：**

一般支持URL编码的库（比如Java中的java.net.URLEncoder）都是按照“application/x-wwwform-urlencoded”的MIME类型的规则进行编码的。实现时可以直接使用这类方式进行编码，把编码后的字符串中加号（+）替换成%20、星号（\*）替换成%2A、%7E替换回波浪号（~），即可得到上述规则描述的编码字符串。

1. 对编码后的参数名称和值使用英文等号（=）进行连接。
2. 再把英文等号连接得到的字符串按参数名称的字典顺序依次使用&符号连接，即得到规范化请求字符串。
3. 对规范化请求字符串使用应用密钥进行签名，将得到的签名值作为signData参数添加到请求header中，即完成对请求签名的过程。

#### 签名示例

