目　录

新手指南
接口规则说明
智能终端开放说明
智能终端开放接口
常见问题
错误码说明-魔蓝
错误码说明-钉钉
智能终端开放接口(非钉)

新手指南

新手指南

新版重要变化

登录入口统一
原版：魔蓝开放接口和钉钉开放接口需要分别注册登录。
新版：统一注册登录路径。

内部应用和**ISV**应用合并为云端接口应用
原版：开发者可申请开通内部应用权限，或ISV应用权限。
新版：统一在工作台-云端接口应用类目下操作，不再区分内部/isv。
老用户登录后仍可切换至原先管理的“内部应用”。

云端应用在线调试更便捷
原版：仅支持魔点平台；接口需要在购买专区申请试用。
新版：支持魔点平台和钉钉平台；
调试界面左下角添加接口，无需申请，即选即用；
调试界面右上角支持一键获取Access Token；
云端APP无需申请发布，调试完毕即可。

开发者资质说明

开发者资质只能通过机构主体申请，暂不支持个人主体认证。

新用户申请

新用户指以下两种情况：
未注册过魔点版开放接口平台和钉钉版开放接口平台的用户；
未注册过魔蓝办公平台的用户。

首次登录魔点开放平台需先注册，使用手机号或者邮箱进行验证。

登录账号后默认进入开发者资质申请页面，填写申请信息包括机构名称、所在地区、详细地址、人员规模、机构类型、联系人、联系电话、企业邮箱、营业执照。
提交资质信息会默认创建一个魔点机构，请确保机构名称与营业执照保持一致，否则无法通过审核。

老用户申请

老用户指以下两种情况：
已注册魔点/钉钉版开放接口平台的用户；
已注册魔蓝办公平台的用户。

没有机构&无开发者资质

登录后默认进入开发者资质申请页面，提交资质信息会默认创建一个魔点机构，请确保机构名称与营业执照保持一致，否则无法通过审核。

已有机构&无开发者资质

登录后选择相应的机构，进入开发者资质申请/修改页面。请确保机构名称与营业执照保持一致，否则无法通过审核。

已有机构&有开发者资质

登录后选择已具备开发者资质的机构，进入开发者资质信息页面，如信息有误，则点击修改，如没有错误则确认提交。

备注：登录页面只展示您的账号具备管理员权限的机构列表。

开发指引

1.云端接口应用-魔点平台

魔点平台：即您的设备激活在魔点平台/魔蓝办公平台。

1.1创建应用

每个机构主体最多支持创建3个魔点APP，您可以用于不同场景APP的开发，或者用于开发环境的隔离。

1.2添加调试机构

创建应用成功后，在应用列表可以获取APPID和APPKey，在相应的APP下添加需要调试的机构信息。
具体请查看接口文档详情。

1.3在线调试

您可根据实际业务需求，点击左下角添加相应的场景业务接口。
通过授权机构获取OrgID，根据接口说明，进行在线调试，调试完毕即可应用于线上。

点击应用名称下方，切换需要调试的机构。

1.4应用详情

进入应用详情页面，可以查看应用授权的机构列表，查看已授权机构的OrgID和OrgKey。

2.云端接口应用-钉钉平台

钉钉平台：即您的设备激活在钉钉平台。

2.1创建应用

每个机构主体最多支持创建3个支持钉钉平台的APP，您可以用于不同场景业务的开发，或者用于开发环境的隔离。

2.2机构授权

点击添加机构，输入的机构钉钉CorpID，申请授权；被申请授权机构的管理员可打开钉钉的“魔点门禁”微应用，点击右上角>更多>授权协议>同意授权。
CorpID获取路径：钉钉开放平台-开发者后台

只有已授权状态的机构可以在线调试。

2.3在线调试

您可根据实际业务需求，点击左下角添加相应的场景业务接口。
根据接口说明，进行在线调试，调试完毕即可应用于线上。

点击应用名称下方，切换需要调试的机构。

2.4应用详情

进入应用详情页面，可以查看应用授权的机构列表，查看已授权机构的OrgID和OrgKey。

3.终端安卓应用

终端安卓应用，指魔点设备提供终端开放能力，包括智能人脸识别能力开放、硬件能力开放、智能语音能力开放等，支持合作伙伴基于开放接口自研的应用。
目前仅支持单机构模式开放，即一个开发者主体仅支持授权一个机构。
已支持的设备类型：D2、D3、MOS745、MOS778、MOS757、MOS968；其它设备如有需求，请联系销售人员。
终端开放使用答疑，请搜索并添加钉钉群号沟通：5440011071。

3.1申请开发权限

点击去申请，输入需要服务的机构钉钉CorpID，申请授权；
被申请机构的管理员可打开钉钉的“魔点门禁”微应用，点击右上角>更多>授权协议>同意授权。
CorpID获取路径：钉钉开放平台-开发者后台

3.2MOS设备管理

可查看此机构下激活的设备信息，具体列表字段含义如下：
【设备名称】：设备激活成功后，用户可自定义设备名称。
【设备序列号（SN 号）】：设备生产序列号。
【设备类型】：魔点设备不同的设备类型代号。
【设备类型名】：魔点设备不同的设备类型代号中文名。
【设备状态】：设备离线 or 在线。
【创建时间】：设备激活时间。
【MOS 设备】：是否为开放版设备，即可集成魔点 SDK 进行自研应用开发的设备。
【安装应用】：设备上成功安装的自研应用。

3.3创建应用

自研应用：仅支持安卓系统的终端应用程序。
进入自研应用列表，点击创建应用，填写应用名称、使用场景和应用描述。
应用创建完成，会生成此应用专属的 Appkey，基于此App key可进行应用的研发、调试、推送至MOS设备和应用升级等操作。
若当前机构下无激活的MOS设备，则无法进行以上操作。

3.4上传应用

同意魔点科技开放平台免责协议后可上传应用文件。
在每次上传应用文件时，您需要先阅读并同意遵守魔点科技开放平台协议文件。

上传应用文件时填写的应用版本信息，需与上传文件内定义的版本号一致，否则会导致上传失败。
目前一个机构下的 MOS 设备，仅支持一款自研应用的安装。请在创建应用及文件上传前确定好目标上传文件。
上传的自研第三方应用文件大小不可超过 100M，且每个应用每天有20次上传版本次数限制。

应用上传成功后，一般会在24小时内审核，请耐心等待。
审核成功后，应用状态栏会显示审核通过。

3.5推送至设备

应用审核成功，若上传的应用适用的设备类型包含MOS设备，可将应用推送至在线的MOS设备。

应用审核失败，则无法推送至设备，需重新上传应用文件。

3.6更新应用

应用文件如有更新，点击更新应用，上传新的应用文件，平台审核成功后，再推送至在线设备。

3.7设备端安装应用

应用成功推送后，设备端在线状态下，当自研应用成功推送后，在线的设备即可会进行应用文件下载与安装。

文件下载成功后即刻会进行应用安装，如下图示。安装成功后，会即刻启动您的自研应用。之后，您就可以在设备上测试使用自研应用了。

3.8元石开放私有化文档

私有化文档链接：元石开放V5.0私有化接口文档

附件

- 元石开放V5.0接口文档.pdf

接口规则说明

接口规则说明

环境准备

jdk
1.7.0_80-b15及以上版本使用。
接入**URL**地址
魔点平台： `https://oapi.moredian.com`
钉钉平台： `https://toapi.moredian.com`

接口规则

接口规则	相关描述
传输方式	仅支持HTTPS传输
URL格式	魔点平台 `https://oapi.moredian.com/uri`；钉钉平台`https://toapi.moredian.com/uri`。每个API都有唯一的`uri`，详见API的接口定义
请求方式	GET/POST
数据格式	大部分请求和返回数据都为JSON格式，有部分请求接口是multipart/form-data。
字符编码	统一采用UTF-8字符编码
请求Headers	HTTP请求中的头信息。大部分设置 `Content-Type:application/json`，部分接口是`multipart/form-data`。
Header	HTTP请求中的头信息中的一个键值对。
Path	HTTP请求路径。如请求地址为 `https://oapi.moredian.com/app/getAppToken`，则Path指`"/app/getAppToken"`。
	请求地址中的参数。如请求地址为 `https://oapi.moredian.com/app/getAp`

	`pToken?appId=XXX&appKey=XXX，则Query为appId:XXX,appKey:XXX。`
Body	指POST请求中body中以键值对存在的参数。

返回值规则

对于API接口调用返回的结果统一采用JSON格式，内容字段采用驼峰式方式命名，所有接口的返回结果里都有result、message，部分接口返回结果中没有data。返回的接口内容格式如下：

```
    {
          "result"://响应码
          "message"://响应消息
          "data"://响应数据
        }
```

开发者需根据result是否为0判断是否调用成功，返回0表示成功，非0表示失败。message仅作参考，后续可能会有变动，因此不可作为是否调用成功的判据。data中返回接口具体内容。
开发者需要特别注意当data为null时的result，以便很好地理解接口返回的信息。
成功调用举例如下：

```
   {
          "result":"0",
          "message":"操作成功",
          "data":true
        }
```

失败调用举例如下：

```
 {
          "result":"-1",
          "message":"操作失败"
        }
```

result为非0时都为错误码。错误码具体释义，参照附录错误码说明。

魔点开放平台**SDK**对接说明

SDK下载

魔蓝版API和钉钉版API分别提供了Java版本SDK供开发者使用，SDK下载：点击下载

开发概述

使用魔点提供的AppId，AppKey以及OrgId，OrgAuthKey（创建机构接口获取或者企业授权后获取）调用SDK对应Request获取token。

请求示例

以下为魔蓝版SDK调用API的请求示例，钉钉版SDK类似，替换接入URL地址即可：

```
// 获取appToken，appToken有效期为返回data的expires，单位秒，过期后需再次获取
IOpenApiClient client = new DefaultOpenApiClient(DefaultProfile.getProfile("https://oapi.moredian.com"));
GetAppTokenRequest request = new GetAppTokenRequest();
request.setAppId(AppId);
request.setAppKey(AppKey);
GetAppTokenResponse response = client.getResponse(request);
```

```
// 获取ISV应用下accessToken，accessToken有效期为返回data的expires，单位秒，过期后需再次获取
IOpenApiClient client = new DefaultOpenApiClient(DefaultProfile.getProfile("https://oapi.moredian.com"));
GetAppOrgAccessTokenRequest request = new GetAppOrgAccessTokenRequest();
request.setAppToken(appToken);
request.setOrgId(OrgId);
request.setOrgAuthKey(OrgAuthKey);
GetAppOrgAccessTokenResponse response = client.getResponse(request);
```

```
// 获取企业内部应用下accessToken，accessToken有效期为返回data的expires，单位秒，过期后需再次获取
IOpenApiClient client = new DefaultOpenApiClient(DefaultProfile.getProfile("https://oapi.moredian.com"));
GetOrgAccessTokenRequest request = new GetOrgAccessTokenRequest();
request.setOrgId(OrgId);
request.setOrgAuthKey(OrgAuthKey);
GetOrgAccessTokenResponse response = client.getResponse(request);
```

```
// 查询内部成员信息
IOpenApiClient client = new DefaultOpenApiClient(DefaultProfile.getProfile("https://oapi.moredian.com"));
GetMemberRequest request = new GetMemberRequest();
request.setAccessToken(accessToken);
GetMemberRequest.GetMemberRequestBody getMemberRequestBody = new GetMemberRequest.GetMemberRequestBody();
getMemberRequestBody.setMemberId(1L);
request.setBody(getMemberRequestBody);
GetMemberResponse response = client.getResponse(request);
```

请求示例说明

1.定义Client对象，设置接口地址URL，Client只需要定义一次即可，其他API复用。

2.构造相关业务Request对象，对应业务的Request对象命名，一般使用驼峰形式+Request，具体请阅读SDK源码注释或者通过路径 “云端接口文档->请求参数->请求示例->Java” 查看。

3.Request对象设置业务参数，其中accessToken、appToken使用GetAppOrgAccessTokenRequest、GetOrgAccessTokenRequest、GetAppTokenRequest等Request对象获取。

4.调用client.getResponse获取执行结果，得到Response对象，与Request相对应。

钉钉开放平台对接

（1）开发者需要与钉钉开放平台对接，以第三方企业应用开发为例，具体操作可参照钉钉开放平台接入指南。https://open.dingtalk.com/document/isv/third-party-enterprise-application-description

（2）对接钉钉开放平台服务端API接口通讯录管理接口，获取基础信息，如用户ID，部门ID等信息。具体接口参照钉钉开放平台服务端API列表。https://open.dingtalk.com/document/isvapp-server/call-server-apis-3

智能终端开放说明

智能终端开放说明

1、智能硬件开放介绍

魔点科技秉持“用户第一、超越边界、共创共赢”的企业价值观，以互联网数字思维，不断优化数字化“武装”，时刻保持数字韧性与敏锐度，实现场景服务在线。开放平台基于魔点智能硬件产品进行打造，致力于协助客户基于魔点提供的人脸识别、门禁考勤、智能语音播报等能力开发场景化的智能应用产品。

2、开发指南

参考智能终端开放接口，基于不同的魔点智能硬件提供的RAM大小和性能差异，建议开发者优化应用运行内存，原则上D2设备保持在100M以内，D3设备保持在200M以内。

3、支持的魔点开放设备（MOS设备）

3.1什么是MOS设备？

MOS设备，即魔点提供终端开放能力的设备，包括智能人脸识别能力开放、硬件能力开放、智能语音能力开放等，支持合作伙伴基于开放接口开发自己的应用APP，以满足不同的场景需求。

3.2目前支持的MOS设备类型如下

MOS设备的具体购买流程，请联络魔点商务渠道人员了解。
SDK版本持续更新，请联系魔点开发人员获取最新版本

系统平台	设备类型	系统版本
Android	D2（D2A）	Android 6.0
Android	D3	Android 9.0
Android	D5	Android 12.0
Android	MOS745	Android 6.0
Android	MOS778	Android 9.0
Android	MOS968	Android 12.0

Android	MOS757	Android 12.0

4、常见问题

4.1 MOS设备如何进行设备网络连接模式切换、连接调试以及获取日志？

购买魔点MOS终端后，若需设备连接调试及获取日志，可在钉钉上公开搜索群“魔点终端开发用户群”（群号：5440011071），描述需求场景并@技术支持申请开放设备连接、查看logcat日志和切换设备网络连接模式（1:host模式，支持无线连接诶，0:device模式，支持有/无线连接）功能。
注：功能申请时需提供设备SN码并确保设备在线

4.2 TTS语音功能如何获取？

目前，钉钉开放终端，可在钉钉微应用“魔点门禁”上“高级权益”中查询购买。
非钉开放终端仅部分设备支持，需联系魔点商务渠道人员和魔点开发人员。

4.3 如何使用扫码和刷卡能力？

D2和MOS745设备，目前不具备能力，若需要则请联络魔点商务渠道人员，单独采购并外接扫码设备。
D3和MOS778、MOS757设备，目前已支持完整的扫码和刷卡能力。
MOS968设备暂不支持扫码能力，若需要需联系魔点商务渠道人员和魔点开发人员。

4.4 MOS终端是否也支持调用云端接口？

终端接口和云端接口并不冲突，都支持。

4.5 推送至设备，提示推送失败，如何解决？

查询设备是否离线，如正常在线，存在软件版本过低的可能，请加入魔点终端用户群（钉钉群号5440011071）联系技术人员处理。

智能终端开放接口

智能终端开放接口

一、更改历史

迭代时间	迭代内容
2022.08.26	base_open.aar 1.0.0-1版本
2022.8.29	修改此文档： 1.部分接口描述修改 2.detectFace接口增加参数，单人识别或多人识别
2022.10.27	base_open_1.0.1_2.aar库更新 changelist： 1.新增appkey授权
2022.11.18	base_open_1.0.2_3.aar库更新 changelist： 1.新增sdk保存log入口
2025.01.16	新增行业智能终端开放

二、系统拓扑

前提条件（必读）

使用业务接口前，需先区分设备所使用环境，钉钉和行业在部分参数调用存在差异，可以包对应的文档为准：
【钉钉终端应用】设备激活方式使用手机钉钉扫码设备上的二维码（手机扫设备蓝牙激活）
【行业终端应用】设备激活方式使用手机扫码生成配置二维码后出示给设备（设备扫手机二维码激活）

三、钉钉终端应用接入指南

3.1、接入流程

1.环境搭建

第三方应用开发者拿到魔点开放sdk后，需要把aar文件置于libs目录下，通过gradle加载开放sdk

2.启动配置

调用开放sdk前需要在AndroidManifest.xml文件中配置启动界面的action，魔点应用会在每次启动后判断第三方应用是否存在，如果存在则会拉起第三方应用

```
<activity android:name=".MainActivity"
        android:exported="true"
        android:launchMode="singleTask">
        <intent-filter>
            <action android:name="android.intent.action.MAIN" />
            <category android:name="android.intent.category.LAUNCHER" />
        </intent-filter>
        <intent-filter>
            <action android:name="com.moredian.open.start" />
            <category android:name="android.intent.category.DEFAULT" />
        </intent-filter>
</activity>
```

3.sdk初始化

调用魔点开放open sdk之前，需要通过MdOpenServer初始化sdk,appKey通过魔点开放平台申请；钉钉终端应用和行业终端应用初始化SDK有些许差异，具体可咨询终端应用对接群；

```
private void initOpenServer() {
        MdOpenServer.getInstance().init(this,this);
        MdOpenServer.getInstance().register(appKey);
}
```

4.数据监听

初始化成功后再去注册数据回流监听

```
@Override
    public void onInitCallBack(boolean isSuccess, String message) {
        Log.d(TAG, "onInitCallBack: isSuccess="+isSuccess+",message="+message);
        isInitSuccess = isSuccess;
        if (isSuccess){
            mOpenListener = new OpenListener();
            MdOpenServer.getInstance().registerListener(mOpenListener);
            getDeviceMode();
            getDeviceBaseInfo();
        }else {
            if (mOpenListener != null){
                MdOpenServer.getInstance().unRegisterListener(mOpenListener);
            }
        }
    }
```

5.sdk注销

应用或者界面销毁，初始化回调失败，需要通过unRegisterListener注销历史的回流监听，否则下次再次注册，会返回多条相同的数据

3.2、业务接口

名词解释

数据回流类**OpenInitListener**

```
private class OpenListener extends IOpenListener.Stub {

        @Override
        public void dataCallBack(String action, String result) throws RemoteException
```

回调方法	参数	类型	描述
dataCallBack	action	String	发起调用open sdk的动作
dataCallBack	result	String	json格式回调

1.sdk初始化

```
/**
     * sdk初始化
     * @param context 系统上下文
     * @param openInitListener 初始化回调
     */
    @Override
    public void init(@NonNull Context context, OpenInitListener openInitListener)
```

方法名	参数	类型	是否必填	描述
init	context	Context	是	上下文
init	openInitListener	OpenInitListener	是	初始化回调

2.sdk注册

```
/**
     * sdk注册
     */
    @Override
    public void register()
```

3.sdk注销

应用销毁后应及时注销sdk，注销后，sdk会断开连接，同时释放持有的资源

```
/**
     * sdk注销
     */
    @Override
    public void release()
```

4.注册数据回流监听

```
/**
* 注册数据回流
* @param listener
*/
@Override
public void registerListener(IOpenListener listener)
```

方法名	参数	是否必填	描述
registerListener	IOpenListener	是	继承自IOpenListener.Stub的回调

5.注销数据回流监听

```
/**
     * 注销数据回流
     * @param listener
     */
    @Override
    public void unRegisterListener(IOpenListener listener)
```

方法名	参数	是否必填	描述
unRegisterListener	IOpenListener	是	继承IOpenListener.Stub的回调

6.获取sdk版本号

```
/**
     * 获取sdk版本号
     * @return
     */
    @Override
    public int getVersionCode()
```

7.获取sdk版本名

```
/**
     * 获取sdk版本名
     * @return
     */
    @Override
    public String getVersionName()
```

8.获取设备信息

```
/**
 * 获取设备基本信息
 * @return
 */
@Override
public String getDeviceBaseInfo()
```

返回结果：

```
{"sn":"090902190730KN0015","romVersion":20124,"romVersionName":"2.1.24.0","appVersion":2158
,"appVersionName":"V1001.2.0.0-xtest"}
```

参数名	类型	描述
sn	String	设备唯一标记
romVersion	long	rom版本号
romVersionName	String	rom版本名
appVersion	int	魔点app版本号
appVersionName	String	魔点app版本名

9.获取设备device host模式

```
 /**
     * 获取设备device host模式
     * @return 1 host模式 0 device模式 -1 获取失败
     */
    @Override
    public int getDeviceHostMode()
```

设备系统每次启动默认处于host模式
device模式：usb线连接adb时设备必须处于device模式
host模式：以太网正常工作时必须处于host模式

10.切换设备device host模式

```
/**
     * 切换设备device host 模式
     * @param model 1 host模式 0 device模式
     */
    @Override
    public void swicthDeviceHostMode(int model)
```

每次切换模式后，因为切换存在延迟，若需要通过getDeviceHostMode获取当前模式，需要延迟1秒以上，否则获取到的状态可能有误

11.开启网络adb

```
/**
     * 是否开启网络adb
     * @param enable true开启 false 关闭
     */
    @Override
    public void enableAdb(boolean enable)
```

12.开启设备调试输出

```
/**
     * 是否设备日志logcat输出
     * @param enable true开启 false 关闭
     */
    @Override
    public void enableLogcat(boolean enable)
```

13.设置刷脸模式

```
/**
     * 设置识别模式
     * @param mode 1多人识别 2单人识别
     */
    @Override
    public void setRecognizeMode(int mode)
```

方法名	参数	类型	是否必填	描述
setRecognizeMode	mode	int	-	1多人识别 2单人识别

14.调用刷脸

```
/**
     * 调用刷脸
     * @param timeOut 刷脸超时时间
     */
    @Override
    public void detectFace(int timeOut)
```

方法名	参数	类型	是否必填	描述
detectFace	timeOut(秒)	int	否	跳转到刷脸页后，超时自动返回三方应用,默认60秒

结果通过IOpenListener回调返回
回调action: detect_face
识别成功json:

```
{"data":{"personId":"10021","personName":"xxx","sceneType":0,"temperature":0.0},"message":"识别成功","result":"200"}
```

识别失败json:

```
{"message":"识别失败，陌生人","result":"400"}
```

15.调用刷卡

```
/**
     * 调用刷卡
     * @param timeOut 刷卡超时时间
     */
    @Override
    public void swipCard(int timeOut)
```

方法名	参数	类型	是否必填	描述
swipCard	timeOut(秒)	int	否	跳转到刷卡页后，超时自动返回三方应用,默认60秒

结果通过IOpenListener回调返回
回调action: swipe_card
成功json:

```
{"data":"2037175851","result":"200","message":"刷卡成功"}
```

失败json:

```
{"message":"设备不支持刷卡","result":"400"}
```

16.调用扫码

```
/**
     * 调用扫码
     * @param timeOut 扫码超时时间
     */
    @Override
    public void scanQrCode(int timeOut)
```

方法名	参数	类型	是否必填	描述
scanQrCode	timeOut(秒)	int	否	跳转到扫码页后，超时自动返回三方应用,默认60秒

回调action: scan_qr_code
成功json:

```
{"data":"www.moredian.com","result":"200","message":"扫码成功"}
```

失败json:

```
{"message":"设备不支持扫码","result":"400"}
```

17.通过卡号获取人员信息

```
/**
     * 通过卡号获取用户信息
     * @param cardNo
     */
    @Override
    public void getUserInfoByCard(String bizNo, @NonNull String cardNo)
```

方法名	参数	类型	是否必填	描述
getUserInfoByCard	bizNo	String	否	业务流水号，需要保证唯一性，设置后魔点开放服务会携带该参数返回
getUserInfoByCard	cardNo	String	是	通过卡号获取用户信息，需要提前在魔点微应用绑定人员和卡号关系

回调action: get_user_info

成功json:

```
{"data":{"bizNo":"xxxxxxxxxxxxx", "cardNum":"2037175851","dingUserId":"1000","id":32909,"name":"xxx","status":1},"message":"query card info success","result":"200"}
```

失败json:

```
{"message":"query card info fail","result":"400"}
```

18.重启设备

```
/**
     * 重启设备
     */
    @Override
    public void reboot()
```

19.开门

```
/**
     * 开门
     */
    @Override
    public void openDoor(boolean isUserDefaultTime, long openDoorTime)
```

方法名	参数	类型	是否必填	描述
openDoor	isUserDefaultTime	boolean	否	是否使用开放服务默认开门时间
openDoor	openDoorTime	Long	否	毫秒，自定义开门时间，isUserDefaultTime false生效

20.设备音量配置

设置了不在0-100范围内的数据，sdk会抛出异常，需要手动捕获

```
/**
     * 设置设备音量
     * @param volume 0-100内整数
     */
    @Override
    public void controlVolume(int volume)
```

方法名	参数	类型	是否必填	描述

controlVolume	volume	int	是	配置设备整体音量大小 范围：0-100

21.设备led补光灯亮度配置

设置了不在0-255范围内的数据，sdk会抛出异常，需要手动捕获

```
/**
     * 设置led补光灯亮度
     * @param value 0-255内整数
     */
    @Override
    public void controlLed(int value)
```

方法名	参数	类型	是否必填	描述
controlLed	value	int	是	配置设备led亮度 范围：0-255

22.设备屏幕亮度配置

设置了不在0-255范围内的数据，sdk会抛出异常，需要手动捕获

```
/**
     * 设置屏幕亮度
     * @param value 0-255内整数
     */
    @Override
    public void controlScreenLight(int value)
```

方法名	参数	类型	是否必填	描述
controlScreenLight	value	int	是	配置设备屏幕亮度 范围：0-255

23.离线语音合成和播放

```
/**
     * 离线语音合成播放
     * @param content
     */
    @Override
    public void playTts(String content)
```

回调action: play_tts
成功json:

```
{"message":"播放成功","result":"200"}
```

失败json:

```
{"message":"TTS组件未初始化","result":"400"}
```

24.无线网络配置

```
/**
     * 无线网络配置
     * @param timeOut 无线网络配置超时时间
     */
    @Override
    public void startWifiSet(int timeOut)
```

方法名	参数	类型	是否必填	描述
startWifiSet	timeOut(秒)	int	否	跳转到无线网配置界面，超时自动返回三方应用,默认60秒

25.有线网络配置

```
/**
     * 有线网络配置
     * @param timeOut 有线网络配置超时时间
     */
    @Override
    public void startEtherSet(int timeOut)
```

方法名	参数	类型	是否必填	描述
startEtherSet	timeOut(秒)	int	否	跳转到有线网配置界面，超时自动返回三方应用,默认60秒

回调action: start_ethernet
成功直接跳转
失败json:

```
{"message":"网线未插入","result":"400"}
```

26.应用日志存储

```
/**
* 保存日志
* @param log 日志信息
*/
@Override
public void saveLog(String log)
```

常见问题

常见问题

Q：开发对接过程中遇到问题如何咨询？

1.非钉业务

打开微信扫码进群

2.钉钉业务

请打开钉钉扫码进群（如果二维码失效请在钉钉使用群号“101610086174”搜索入群。）

Q：如何添加钉钉平台的机构**CorpID**实现授权绑定？

1.在手机钉钉工作台中搜索【魔点门禁】微应用并开通；登录钉钉开放平台-开发者后台，获取CorpID。

2.在开放平台应用详情，点击添加机构；输入CorpID，提交申请授权。

3.机构管理员打开手机钉钉上的“魔点门禁”微应用，点击右上角>更多>授权协议>同意授权。

Q：提示**“**没有接口调用权限**”**，如何解决？
(1)请检查请求路径是否有误：例如大小写错误、字符错误等。
(2)如请求路径没有问题，请确认是否已申请该接口权限或者确认该接口是否已到期。

Q：如何使用回调接口？
先注册回调，回调类型取决于实际需求，回调地址需支持公网访问。
以人脸识别开门记录回调为例，注册回调的类型为 REC_SUCCESS；实际产生刷脸开门记录后，我们会访问您注册的回调地址，请求传送相应的信息
【人脸识别开门记录回调】调用流程

【访客预约记录回调说明】调用流程

Q：没收到回调信息如何排查？
存在以下三种可能性：
(1)没有注册回调，即未提供回调类型和回调地址。
(2)回调接口注册的callbackTag与实际需要触发的业务不符。比如实际业务要识别回调，需注册“REC_SUCCESS”，但注册了其它类型。
(3)回调接口注册的callbackUrl地址无法访问，请确认。如还未解决，请加入魔点开发者交流群（钉钉群号34405002515）联系技术人员处理。

Q：回调信息是否有重试机制？
当业务消息回调给用户的callbackUrl 未成功，魔点系统会自动触发周期为11次、时长为24小时的重试机制，间隔为1分钟, 1分钟, 3分钟, 5分钟, 20分钟, 30分钟, 1小时, 2小时, 4小时, 8小时, 8小时。

Q：回调信息验签机制是怎么样的？
选择是否验签，并不影响业务回调的信息内容。
魔点提供了回调签名校验的机制如下：
（1）计算用户云端本地签名值：
signature = BASE64(HMACSHA1(nonce + orgId + signVersion + timestamp + bodyMd5, orgAuthKey))

（2） 参数说明：
其中nonce、orgId、signVersion、timestamp 通过魔点云端发送的回调请求Query参数中获取；
其中bodyMd5 = MD5(bodyStr)，MD5为MD5加密32位小写。bodyStr为魔点云端发送的回调请求中Body报文里原始字符串，需要直接将body报文转成String，不能进行Json反序列化之后再转换成String，否则会出现数据中字段顺序不一致的情况，从而导致验签不通过；body报文转成String。
Java语言参考
其中orgAuthKey为orgId对应的orgAuthKey，也就是获取accessToken对应的orgAuthKey。
（3）签名比对：
用户云端本地签名值signature计算完成，与魔点云端发送的回调请求Query参数中获取的签名值signature进行比较，一致则验签通过，不一致则验签未通过。

Q：如何调用三方鉴权接口？

Q：调用三方鉴权接口自定义文案，语音通报有效果，屏显没有效果是什么原因
存在设备包版本过低的可能，请先在设备端操作升级，如升级失败或无法升级，再联系售后人员。

Q：接口中人脸图片如何上传？
参数类型是File，参数值是一个图片，二进制文件，需要用post multipart/form-data的方式上传。

Q：接口中人脸图片有何限制？
人脸图片大小不能超过2M，图片的像素需要小于1920X1920。
人员和成员相关接口的人脸图片，人脸图片中人脸的像素大小至少为200X200，并非整体图片像素大小。

Q：元石开放私有化文档

私有化文档附件：元石开放V5.0私有化接口文档

附件

- 元石开放V5.0接口文档.pdf

错误码说明-魔蓝

错误码	描述信息
010000001	效参数异常错误
010000006	服务调　超时
010170022	token　效
011190400	效参数
011190403	没有权限
011190404	未找到资源
011190500	服务器异常
011191500	远程服务调　失败
011191404	权限组不存在
011199404	员与权限组　绑定关系
011191403	权限组绑定类型异常
011192403	权限组同步中
011193403	全员组已满员，　需绑定操作
011192404	员不存在
011190001	员未录　　脸
011197404	部　不存在
011193404	部　下不存在成员
011194404	成员未存在部　中
011195404	群组　绑定部
011196404	部　与权限组　绑定关系
011198404	设备不存在
011190002	更新tpId失败
011190003	期格式错误

011190004	会议室初始化配置失败
011190005	图/　件超出允许范围
011190006	设备未开启会议室模式
011190007	期范围错误
011190008	未检测到　脸
011190009	脸超出最　检测范围
011190010	[%s]与　禁业务互斥
011190011	[%s]与考勤业务互斥
011190012	[%s]与会议室业务互斥
011190013	[%s]与访客业务互斥
011190014	[%s]与签到业务互斥
011190015	[%s]与测温业务互斥
011190016	[%s]与梯控业务互斥
011190017	部　或楼宇名称　度不能超过100位
011190018	楼宇名称不能为空
011190019	请输　正确的单元数
011190020	请输　正确的楼层数
011190021	请输　正确的户室数
011500500	服务器异常
011500400	效参数
011500001	验证码错误
011500002	验证码已过期
011500003	账号不存在
011500004	机号格式错误
011500005	账号已存在

010990000	服务器繁忙，请稍后重试
010990001	该时段已被抢占
010990003	会议室时间有误
010990004	会议时间不在会议室开放时间段内
010990008	预约时段超过会议室单次可预定最　时间
010990011	会议时间不在会议室开放时间段内
010990012	会议预约不存在
010990016	基础配置 code不可
010990018	会议室已被删除，请重新预约
010990019	会议室已停　使
010990028	前会议模式只　持参会　员
010990030	会议预约时间有误，请重新选择
010990033	会议室开放时间不合法
010990034	会议的开始时间不能　于当前时间
010420001	效参数异常错误
010421002	外部联系　已存在
010421003	外部联系　不存在
010421014	外部联系[%s]　已存在，　需重复添加
010421017	tpId已存在于外部联系
010421020	外部联系[%s]　已存在，　需重复添加
010421021	身份证号码已存在于外部联系
010421022	外部联系[%s] [%s]　已存在，　需重复添加
010423001	机构已存在
010423002	机构不存在
010423003	机构尚未停
010423004	业务尚未完全停

010423005	未找到根位置
010423006	位置不存在
010423007	位置不存在
010423008	拒绝删除树型位置节点
010423009	部　已存在
010423010	部　不存在
010423011	上级部　不存在
010423012	组已存在
010423013	组已不存在
010423014	拒绝修改系统默认群组
010423015	拒绝删除系统创建的组
010423016	系统默认群组不　持本操作
010140020	脸与[%s]相似
010423021	机构不存在
010423022	根部　不能被删除
010423023	前　次变更操作还未完成，稍后再试
010423025	部tpId　已存在
010423026	部　未授权
010423027	权限组tpId已存在
010423028	权限组同步中
010140001	参数有误
010140002	成员[%s]已存在
010140003	成员不存在
010140004	存储图　异常
010140005	存储图　失败

010140006	请上传合格的　脸图
010140007	识别照　中存在多张　脸
010140010	裁剪　脸照　异常
010140011	成员关系不存在
010140012	部　下还有成员存在
010140013	部　下还有　部　存在
010140014	机号已存在于机构成员
010140015	脸照　太
010140016	管理员　法被删除
010140017	tpId[%s]的成员已存在
010140019	身份证号码有误
010140021	员未授权
010140023	脸　　少为%s*%s
010140024	号已存在
010140026	图　　法访问
010140027	机号格式不正确
010140028	邮箱格式不正确
010140029	身份证格式不正确
010140030	超出　号　持最　位数
010140031	邮箱已存在于机构成员
010140050	机号码不正确
010141005	员存在群组关系
010141006	员尚未上传　脸照
010141008	脸照　不完整
010141009	眼部被遮挡
010141010	墨镜遮挡眼部

010141011	嘴巴被遮挡
010141012	脸旋转　度过
010141013	脸上下　度过
010141014	脸左右　度过
010141015	照　太模糊
010141016	照　不符合要求
010141017	不是　脸
010141019	开　信号值只能是0到8位数字
010141020	脸库变更频繁
010141021	脸图　　正脸
010141022	脸图　旋转度过
010141023	带眼镜且已闭眼
010141024	未带眼镜且已闭眼
010141025	卡已经绑定
010141026	员已绑卡
010141027	卡已冻结
010141028	卡被注销
010141029	效卡
010141030	卡信息不存在
010141031	未绑卡,操作　效
010141032	卡　已冻结，请勿重复操作
010141033	该机构下　可导出的　员
010141034	选择　数超过1万　，请调整
010141035	请核对10位卡号是否正确
010141036	sip账号已存在

010141037	sip密码加密失败
010141038	楼宇信息不存在
010141039	数据需要导出
010141040	导出任务异常
010141041	此卡已被绑定
010141042	卡号格式错误，必须为10位数字[%s]
010141043	此卡已被解绑
010423029	部　名称重复
010423030	批量创建部　数量上限1000个
010142000	脸亮度过低
010142001	脸佩戴　罩
010142002	脸完整度过低
010142003	脸被遮挡
010142004	眼睛闭合
010142005	嘴巴未闭合

错误码说明-钉钉

错误码	描述信息
010000001	无效参数异常错误
010000006	服务调用超时
010170022	Token 无效
011190400	无效参数
011190403	没有权限
011190404	未找到资源
011190500	服务器异常
011191500	远程服务调用失败

智能终端开放接口(非钉)

一、非钉终端应用接入指南

需要接入者具备基本的android开发者专业能力标准即可完成以下对接，以下文字输出涉及开发相关专业技术知识

该sdk需要配套魔点最新的MOS版本使用(关于设备版本可询问技术支持确认)

更改历史

迭代时间	迭代内容
2026.09.24	文档修订

1.环境搭建

1.1 sdk集成(环境搭建)

第三方应用开发者拿到demo中内置的魔点开放sdk后，需要把aar文件置于自身android项目中的**libs**目录下，通过**gradle**加载这个开放sdk

sdk 版本会持续更新 但sdk前缀格式固定，具体可参考最新版本demo，并采用demo内置的sdk文件进行集成

1.2 sdk引入

在项目中build.gradle的dependencies添加如下代码

implementation fileTree(include: [‘*.aar’], dir: ‘libs’)
name 与aar的名称保持一致，这两个aar都需要放入同一层级

1.3 NDK依赖

build.gradle中添加

1.4 第三方依赖导入

build.gradle中添加

基础SDK有使用到该三方类库网络请求组件/greendao数据库/collections库依赖，请开发者集成上述依赖到自身项目中

2.基础开放能力配置

2.1启动配置

调用开放sdk前需要在AndroidManifest.xml文件中配置启动界面的action，魔点应用会在每次启动后判断第三方应用是否存在，如果存在则会拉起第三方应用

```
1.  //示范例子(第三方应用首页)
2.   <activity android:name=".MainActivity" android:exported="true" android:launchMode="singl
eTask">
3.     <intent-filter>
4.       <action android:name="android.intent.action.MAIN" />
5.       <category android:name="android.intent.category.LAUNCHER" />
6.     </intent-filter>
7.     <intent-filter>
8.       <action android:name="com.moredian.open.start" />
9.       <category android:name="android.intent.category.DEFAULT" />
10.    </intent-filter>
11.  </activity>
```

上述代码标红的部分为第三方应用开发者需要关注的点，在自己开发的应用项目中的主页面(首页)添加的代码

2.2初始化(MdOpenServer)

调用魔点开放open sdk之前，需要通过MdOpenServer初始化sdk,需传入appId和appKey，可通过魔点开放平台->终端安卓应用->行业终端应用-创建项目（应用）获取

```
1.  private void initOpenServer() {
```

```
    MdOpenServer.getInstance().init(this, this);
}
```

2.3数据监听

初始化成功后再去注册数据回流监听

```
@Override
public void onInitCallBack(boolean isSuccess, String message) {
    Log.d(TAG, "onInitCallBack: isSuccess=" + isSuccess + ",message=" + message);
}
```

2.4 SDK注销

应用或者界面销毁，初始化回调失败，需要通过_unRegisterListener_注销历史的回流监听，否则下次再次注册，会返回多条相同的数据

```
MdOpenServer.getInstance().unRegisterListener(callback);
```

2.5业务接口名词解释

OpenInitListener 初始化回调

```
public interface OpenInitListener {
    void onInitCallBack(boolean isSuccess, String message);
}
```

回调方法	参数	类型	描述
onInitCallBack	isSuccess	boolean	sdk初始化是否成功
	message	string	返回__message

AuthCallBack 授权回调

```
public interface AuthCallBack {
    void onAuthBack(boolean var1, String var2);
}
```

回调方法	参数	类型	描述
onAuthBack	isSuccess	boolean	授权是否成功
	message	string	授权结果信息

OpenResultListener 数据回调

```
public interface OpenResultListener {
  void resultCallBack(String action, String result);
  void onRecognizeWithCorpFace(String result, ParcelFileDescriptor pfd);
}
```

回调方法	参数	类型	描述
resultCallBack	*action*	*string*	调用的开放方法
	result	*string*	返回结果
onRecognizeWithCorpFace	*result*	*string*	识别回调*(json)*
	pfd	*ParcelFileDescriptor*	识别人脸数据

sdk初始化

```
public void init(@NonNull Context context, OpenInitListener openInitListener)
```

方法名	参数	类型	是否必填	描述
init	*context*	*Context*	是	上下文
	openInitListener	*OpenInitListener*	是	初始化回调

获取授权

```
public void getAuth(String appId, String appKey, AuthCallBack authCallBack)
```

方法名	参数	类型	是否必填	描述
getAuth	*appId*	*String*	是	终端开放平台创建的*appId*
	appKey	*String*	是	终端开放平台创建的*appKey*
	authCallBack	*AuthCallBack*	是	

sdk注销

应用销毁后应及时注销sdk，注销后，sdk会断开连接，同时释放持有的资源

```
public abstract void release();
```

注册数据回流监听

```
1. public abstract void registerListener(OpenResultListener listener);
```

方法名	参数	是否必填	描述
registerListener	*OpenResultListener*	是	注册回调监听

注销数据回流监听

```
1. public abstract void unRegisterListener(OpenResultListener listener);
```

方法名	参数	是否必填	描述
unRegisterListener	*OpenResultListener*	是	注册回调监听

获取**sdk**版本号

```
1. public abstract int getVersionCode();
```

获取**sdk**版本名

```
1. public abstract String getVersionName();
```

获取设备信息

```
1. public abstract String getDeviceBaseInfo();
```

返回结果：
{“sn”:”090902190730KN0015”,”romVersion”:20124,”romVersionName”:”2.1.24.0”,”appVersion”:2158,”appVersionName”:”V1001.2.0.0-xtest”}

参数名	类型	描述
sn	*string*	设备唯一标识
romVersion	*long*	*rom*版本号
romVersionName	*string*	*rom*版本名

appVersion	*int*	魔点*app*版本号
appVersionName	*string*	魔点*app*版本名

获取设备**device host**模式

```
1. public abstract int getDeviceHostMode();
```

返回结果:
return 1 host模式 0 device模式 -1 获取失败
设备系统每次启动默认处于host模式
device模式：usb线连接adb时设备必须处于device模式
host模式：以太网正常工作时必须处于host模式

切换设备**device host**模式

```
1. public abstract void switchDeviceHostMode(int model);
```

方法名	参数	类型	是否必填	描述
switchDeviceHostMode	*model*	*int*	是	设置模式 *(1 host*模式 ***0 device***模式*)*

每次切换模式后，因为切换存在延迟，若需要通过getDeviceHostMode获取当前模式，需要延迟*1*秒以上，否则获取到的状态可能有误

开启网络**adb**

```
1. public abstract void enableAdb(boolean enable);
```

方法名	参数	类型	是否必填	描述
enableAdb	*enable*	*boolean*	是	*true*开启 *false* 关闭

开启设备调试输出

```
1. public abstract void enableLogcat(boolean enable);
```

方法名	参数	类型	是否必填	描述
enableLogcat	*enable*	*boolean*	是	*true*开启 *false* 关闭

设置刷脸模式

```
1. public abstract void setRecognizeMode(int mode);
```

方法名	参数	类型	是否必填	描述
setRecognizeMode	*model*	*int*	是	*1*多人识别 *2*单人识别

调用刷脸

```
1. public abstract void detectFace(int timeOut);
```

方法名	参数	类型	是否必填	描述
detectFace	*timeOut*	*int*	是	跳转到刷脸页后，超时自动返回三方应用,默认*60*秒

结果通过_IOpenListener回调返回，详细可参考demo

回调action:detect_face

识别成功json:
{“data”:{“personId”:”10021”,”personName”:”xxx”,”userType”:1},”message”:”识别成功”,”result”:”200”}

识别失败json:
{“message”:”识别失败，陌生人”,”result”:”400”}

调用刷卡

```
1. /**
2.  * 调用刷卡
3.  * @param timeOut 刷卡超时时间
4.  */
5. @Override
6. public void swipCard(int timeOut)
```

方法名	参数	类型	是否必填	描述
swipCard	timeOut(秒)	int	否	跳转到刷卡页后，超时自动返回三方应用,默认60秒

结果通过IOpenListener回调返回
回调action: swipe_card
成功json:

{“data”:”2037175851”,”result”:”200”,”message”:”刷卡成功”}

失败json:

{“message”:”设备不支持刷卡”,”result”:”400”}

刷卡回调

{“data”:”0123456789”,”message”:”刷卡成功”,”result”:”200”}

调用扫码

```
1. public void scanQrCode(int timeOut)
```

方法名	参数	类型	是否必填	描述
scanQrCode	*timeOut(秒)*	*int*	否	跳转到扫码页后，超时自动返回三方应用,默认*60*秒

回调action:scan_qr_code

成功json:

{“data”:”www.moredian.com”,”result”:”200”,”message”:”扫码成功”}

失败json:

{“message”:”设备不支持扫码”,”result”:”400”}

通过卡号获取人员信息

```
1. public abstract void getUserInfoByCard(@NonNull String cardNo);
```

方法名	参数	类型	是否必填	描述
getUserInfoByCard	*cardNo*	*string*	是	通过卡号获取用户信息，需要提前在魔点微应用绑定人员和卡号关系

回调action:get_user_info

成功json:

{“data”:
{“cardNum”:”2037175851”,”dingUserId”:”1000”,”id”:32909,”name”:”xxx”,”status”:1},”message”:”query card info success”,”result”:”200”}

失败json:

{“message”:”query card info fail”,”result”:”400”}

重启设备

```
1. public abstract void reboot();
```

开门

```
1. public abstract void openDoor(boolean isUserDefaultTime, long openDoorTime);
```

方法名	参数	类型	是否必填	描述
openDoor	*isUserDefaultTime*	*boolean*	是	是否使用开放服务默认开门时间
	openDoorTime	*long*	是	毫秒，自定义开门时间，*isUserDefaultTime false*生效

设备音量配置

设置了不在0-100范围内的数据，sdk会抛出异常，需要手动捕获

```
1. public abstract void controlVolume(int volume);
```

方法名	参数	类型	是否必填	描述
controlVolume	*volume*	*int*	是	配置设备整体音量大小*0~100*

返回第三方应用

```
1. public abstract void back();
```

设备led补光灯亮度配置

设置了不在0-255范围内的数据，sdk会抛出异常，需要手动捕获

```
1. public void controlLed(int value)
```

方法名	参数	类型	是否必填	描述
controlLed	*value*	*int*	是	配置设备整体***led***亮度大小***0~255***

设备屏幕亮度配置

设置了不在0-255范围内的数据，sdk会抛出异常，需要手动捕获

```
1. public void controlScreenLight(int value)
```

方法名	参数	类型	是否必填	描述
controlScreenLight	*value*	*int*	是	配置设备整体屏幕亮度***0~255***

无线网络配置

```
1. /**
2.      * 无线网络配置
3.      * @param timeOut 无线网络配置超时时间
4.      */
5.     @Override
6.     public void startWifiSet(int timeOut)
```

方法名	参数	类型	是否必填	描述
startWifiSet	timeOut(秒)	int	否	跳转到无线网配置界面，超时自动返回三方应用,默认60秒

有线网络配置

```
1. /**
2.      * 有线网络配置
3.      * @param timeOut 有线网络配置超时时间
4.      */
5.     @Override
6.     public void startEtherSet(int timeOut)
```

方法名	参数	类型	是否必填	描述
startEtherSet	timeOut(秒)	int	否	跳转到有线网配置界面，超时自动返回三方应用,默认60秒

回调action: start_ethernet
成功直接跳转
失败json:
{“message”:”网线未插入”,”result”:”400”}

AI-Camera配置

仅MOS968支持
/**

```
 * 调用启用/注销监听
 * @param enable  注册/注销
 */
@Override
public void enableAiCamera(boolean enable)
```

方法名	参数	类型	是否必填	描述
enableAiCamera	enable	boolen	是	需要ai-camera监听则传true，否则传false及时注销

结果通过IOpenListener回调返回
回调action: enable_ai_camera
成功json:

{“data”:0,”message”:”获取AI-Camera数据”,”result”:”200”}

失败json:

{“message”:”开放接口不支持”,”result”:”400”}

AI-Camera回调

仅MOS968支持

```
action: enable_ai_camera
result: {"data":1,"message":"获取AI-Camera数据","result":"200"}
```

3.问题排查及日志抓取

3.1 日志采集/提供

sdk内置异常日志采集分析能力，如您集成**sdk**后 设备在调试/日常运行过程中遇到相关问题，可提供相关日志并联系我们开发进行分析答疑解惑。**sdk**日志占比存储空间极小，且内置自动清除能力，无需担心**sdk**日志过大导致设备载荷异常

日志目录在设备sd卡外部存储目录下面，请不要随意清除该部分日志
/sdcard/Android/data/XXXXX/files/logs/MoreDian

XXXXX为您开发的应用的具体包名下

3.2 sdk database获取(如无则无需提供)

/data/data/您的应用包名**/databases/faceJurisdiction.db**