1. 开发环境配置

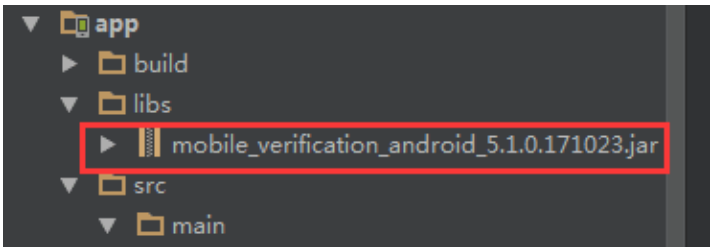
sdk技术问题沟通QQ群：609994083

1.1. 总体使用流程

1. 调用SDK方法来获得`token`，步骤如下：
   a. 构造SDK中认证工具类AuthnHelper的对象；
   b. 使用AuthnHelper中的getToken方法，获得token。
2. 使用平台本机号码校验接口，进行号码校验

1.2. 新建工程并导入SDK的jar文件

将`mobile_verification_android_*.jar`拷贝到应用工程的libs目录下，如没有该目录，可新建；



1.3. 配置AndroidManifest

注意：为避免出错，请直接从Demo中复制带标签的代码

配置权限

```
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.READ_PHONE_STATE" />
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />
<uses-permission android:name="android.permission.SEND_SMS" />
<uses-permission android:name="android.permission.CHANGE_NETWORK_STATE" />
<uses-permission android:name="android.permission.WRITE_SETTINGS"/>

```

通过以上步骤，工程就已经配置完成了。接下来就可以在代码里使用统一认证的SDK进行开发了

1.4. SDK使用步骤

1. 创建一个AuthnHelper实例

`AuthnHelper`是SDK的功能入口，所有的接口调用都得通过`AuthnHelper`进行调用。因此，调用SDK，首先需要创建一个`AuthnHelper`实例，其代码如下：

```
public void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    mContext = this;
    ……
    mAuthnHelper = AuthnHelper.getInstance(mContext);
    }
```

2. 实现回调

所有的SDK接口调用，都会传入一个回调，用以接收SDK返回的调用结果。结果以`JsonObjent`的形式传递，`TokenListener`的实现示例代码如下：

```
mListener = new TokenListener() {
    @Override
    public void onGetTokenComplete(JSONObject jObj) {
        if (jObj != null) {
            mResultString = jObj.toString();
            mHandler.sendEmptyMessage(RESULT);
            if (jObj.has("token")) {
                mtoken = jObj.optString("token");
            }
        }
    }
};
```

3. 接口调用

```
mAuthnHelper.getToken(Constant.APP_ID,
        Constant.APP_KEY,
        mListener);
```

2. SDK方法说明

2.1. 获取管理类的实例对象

2.1.1. 方法描述

获取管理类的实例对象

原型

```
1  public AuthnHelper (Context context)
```

2.1.2. 参数说明

参数	类型	说明
context	Context	调用者的上下文环境，其中activity中this即可以代表。

2.2. 获取校验凭证token

2.2.1. 方法描述

开发者向统一认证服务器获取用户身份标识`openId`和临时凭证`token`。

openId：每个APP每个手机号码对应唯一的openId。

临时凭证token：开发者服务端可凭临时凭证token通过3.1本机号码校验接口对本机号码进行验证。

原型

```
1  public void getToken(final String appId,
2              final String appKey,
3              final TokenListener listener)
```

2.2.2. 参数说明

请求参数

参数	类型	说明
appId	String	应用的AppID
appkey	String	应用密钥
listener	TokenListener	TokenListener为回调监听器，是一个java接口，需要调用者自己实现；TokenListener是接口中的认证登录token回调接口，OnGetTokenComplete是该接口中唯一的抽象方法，即void OnGetTokenComplete(JSONObject jsonobj)

响应参数

OnGetTokenComplete的参数JSONObject，含义如下：

字段	类型	含义
resultCode	String	接口返回码，“103000”为成功。具体响应码见4.1. 本机号码校验接口返回码
authType	String	认证类型：0:其他； 1:WiFi下网关鉴权； 2:网关鉴权； 3:短信上行鉴权； 7:短信验证码登录
authTypeDes	String	认证类型描述，对应authType
resultDesc	String	失败时返回：返回错误码说明
token	String	成功时返回：临时凭证
openId	String	成功时返回：用户身份唯一标识

2.2.3. 示例

请求示例代码

```
mAuthnHelper.getToken(Constant.APP_ID,
        Constant.APP_KEY,
        mListener);
```

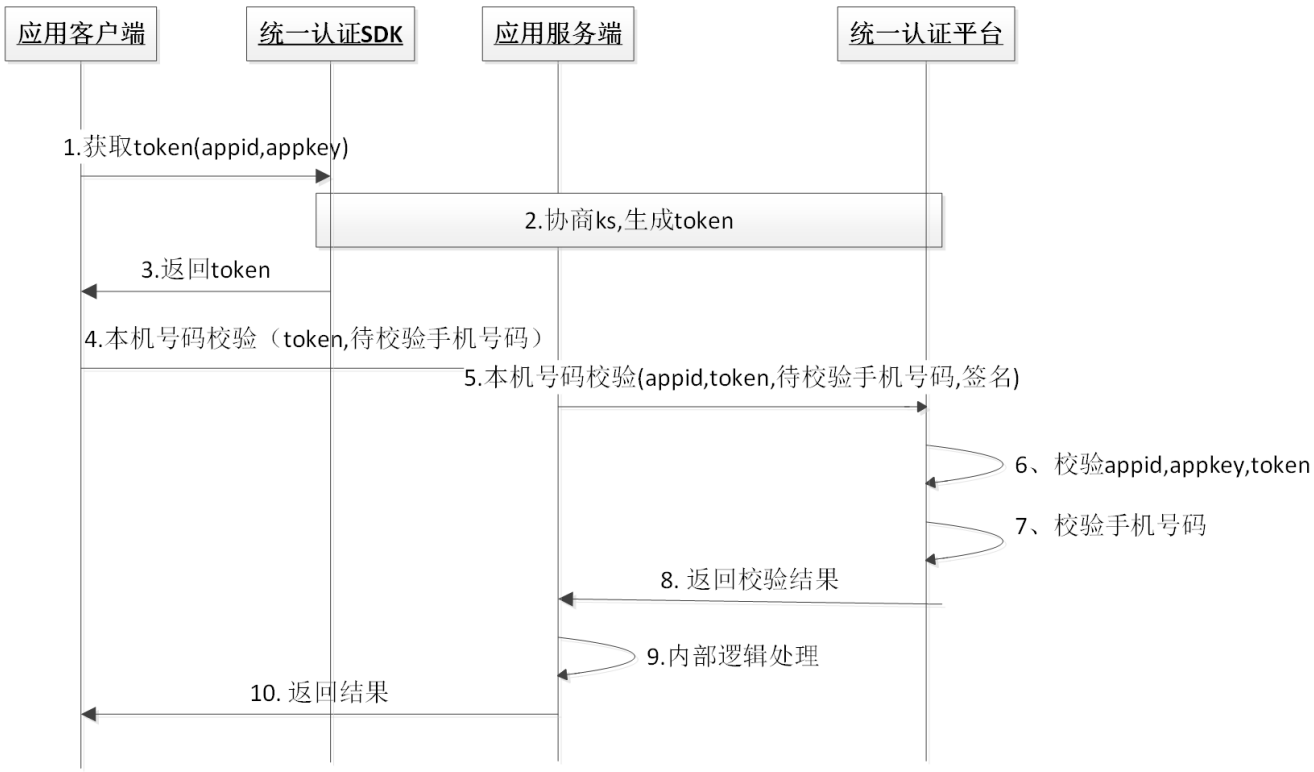
响应示例代码

```
{
    "resultCode": "103000",
    "authType": "2",
    "authTypeDes": "网关鉴权",
    "openId": "9M7RaoZH1Q95QzY99YFkeFDO4xDfOv5q4BVlwn_0zJNNlNYUkxrw",
    "token":
"848401000133020037 4D455979526A49354E6A59774E444D314E454E47516B4D3140687474703A
2F2F3231312E3133362E31302E3133313A383038302F40303103000402D59A6B040012383030313
230313730383138313031343437050010D2F28C555CB54316B7D031DE9F6F6B1EFF0020F07B4AAF
C3B1499A250AAAB4272BBFB565B440FFA5C8257E90C28595956CC224"
}
```

3. 平台接口说明

3.1. 本机号码校验接口

3.1.1. 业务流程



3.1.2. 接口说明

校验用户输入的号码是否本机号码。 应用将手机号码传给统一认证SDK，统一认证SDK向统一认证服务端发起本机号码校验请求，统一认证服务端通过网关或者短信上行获取本机手机号码和第三方应用传输的手机号码进行校验，返回校验结果。

调用次数说明：本产品属于收费业务，开发者未签订服务合同前，每天总调用次数有限。

请求地址： https://www.cmpassport.com/openapi/rs/tokenValidate

协议： HTTPS

请求方法： POST+json

回调地址：请参考开发者接入流程文档

3.1.3. 参数说明

请求参数

参数	类型	层级	约束	说明
header		**1**	必选	
version	string	2	必选	版本号,初始版本号1.0,有升级后续调整
msgId	string	2	必选	使用UUID标识请求的唯一性
timestamp	string	2	必选	请求消息发送的系统时间，精确到毫秒，共17位，格式：20121227180001165
appId	string	2	必选	应用ID
body		**1**	必选	
openType	String	2	否，requestertype字段为0时是	运营商类型: 1:移动; 2:联通; 3:电信; 0:未知
requesterType	String	2	是	请求方类型: 0:APP; 1:WAP
message	String	2	否	接入方预留参数，该参数会透传给通知接口，此参数需urlencode编码
expandParams	String	2	否	扩展参数格式：param1=value1|param2=value2 方式传递，参数以竖线 | 间隔方式传递，此参数需urlencode编码。
phoneNum	String	2	是	待校验的手机号码的64位sha256值，字母大写。（手机号码+appKey+timestamp）（注：“+”号为合并意思）
token	String	2	是	身份标识，字符串形式的token

参数	类型	层级	约束	说明
sign	String	2	是	签名，HMACSHA256(appId+msgId+phonNum+timestamp+token+version)，输出64位大写字母 （注：“+”号为合并意思，不包含在被加密的字符串中,appkey为秘钥, 参数名做自然排序（Java是用TreeMap进行的自然排序））

请求示例

```
{
  "body": {
    "openType": "1",
    "requesterType": "1",
    "message ": "",
    "expandParams": "",
    "phoneNum":
 "4526285940b6fa7fef49e1dcb04ee944f41a8745444015daf2771bfb7ad7c800",
    "token": "",
    "sign": ""
    },
  "header": {
    "msgId ": "61237890345",
    "timestamp ": "20160628180001165",
    "version ": "1.0",
    "appId ": "0008"
  }
}
```

响应参数

参数	层级	类型	约束	说明
header	**1**		必选	
msgId	2	string	必选	对应的请求消息中的msgid
timestamp	2	string	必选	响应消息发送的系统时间，精确到毫秒，共17位，格式：20121227180001165

参数	层级	类型	约束	说明
appId	2	string	必选	应用ID
resultCode	2	string	必选	规则参见具体接口返回码说明
body	**1**		必选	
resultDesc	2	String	必选	返回结果描述信息: 000:是本机号码 001:非本机号码 102:参数无效 108:无效的手机号 302:签名校验不通过 606:token校验失败 999:系统异常 102315：使用次数为0 其中，000和001状态纳入计费次数
message	2	String	否	接入方预留参数，该参数会透传给通知接口，此参数需urlencode编码
expandParams	2	String	否	扩展参数格式：param1=value1|param2=value2 方式传递，参数以竖线 | 间隔方式传递，此参数需urlencode编码。

响应示例

```
{
  "body": {
    "resultDesc ": "",
    "message": "",
    "expandParams ": ""
    },
   "header": {
    "msgId": "61237890345",
    "timestamp": "20160628180001165",
    "resultCode": "1.0"
  }
}
```

4. 平台返回码说明

4.1. 本机号码校验接口返回码

本返回码表仅针对`本机号码校验接口`使用

返回码	说明
000	成功
001	接口处理失败
002	短信验证码下发失败
003	生成短信验证码失败
004	短信验证码下发频率限制
026	同一用户认证过于频繁
028	与上次登陆地点不一致，两次登陆地点不一样
101	接口名称无效
102	参数无效
103	短信验证码已过期
104	短信验证码错误
105	密码错误
106	ip地址非法
107	免登录令牌无效
108	手机号码格式错误/无效的手机号
109	邮箱地址格式错误
110	新密码不能与当前密码相同

返回码	说明
111	登录认证类型不存在
112	登录认证类型非法
113	WAP网关获取信息失败
200	用户已开通通行证，但未开通当前业务
201	用户已开通通行证，且已开通当前业务
202	短信网关异常
203	数据库异常
204	访问频率限制
205	调用app接口失败
206	该用户不存在
207	用户还未创建通行证
208	UID不存在或失效
209	UID与openid不匹配
210	二级密码输入错误
211	密码加密的哈希类型错误
212	上行短信为空或发送失败
228	解码失败
233	接收联通取号结果为空
240	接收电信取号结果为空
302	签名校验不通过
303	参数解析错误

返回码	说明
601	公钥不存在或过期
602	用户密码解密失败
603	客户端随机数解密失败
604	用户未登录或密钥不存在
605	用户密钥已过期
606	验证Token失败
607	强制重新登录
608	clientid不存在
609	Android包名或签名值验证错误
610	身份凭证不存在（token）
611	imsi验证失败
612	imei验证失败
999	未知系统异常
102315	使用次数为0

4.2. SDK返回码说明

错误编号	返回码描述
103000	成功
102101	无网络
102102	网络异常

错误编号	返回码描述
102223	数据解析异常
102121	用户取消认证
102505	业务未注册
102506	请求出错
102507	请求超时
102201	自动登陆失败
102202	应用签名失败
102203	输入参数错误
102204	正在gettoken处理
102210	指定号码非本机号码
102211	短信验证码验证成功后返回随机码为空
102222	http响应头中没有结果码
102299	other failed
102302	调用service超时
103117	mac异常 macError
103200	ks无需更新
103203	缓存用户不存在
200001	imsi为空，跳到短信验证码登录
200002	imsi为空，没有短信验证码登录功能
200003	复用中间件首次登录
200004	复用中间件二次登录

错误编号	返回码描述
200005	用户未授权READ_PHONE_STATE
200006	用户未授权SEND_SMS
200007	不支持的认证方式 跳到短信验证码登录
200008	不支持的认证方式 没有短信验证码登录功能
200009	应用合法性校验失败
200010	imsi获取失败或者没有sim卡，预取号失败