# BJCA 医网签 SDK 集成文档

# ——Android（2.1.3）

二〇一八年 四月

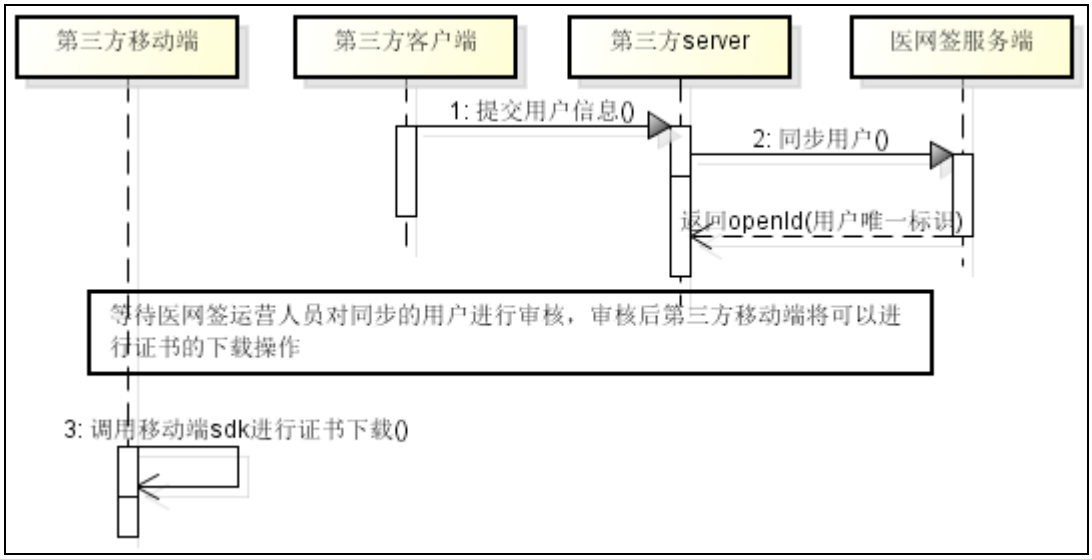
北京数字认证股份有限公司
北京市海淀区北四环西路 68 号左岸公社 15 层
TEL：86-10-58543524

# 1 移动端集成说明

医网签通过身份认证和电子签名保证医疗体系的信息安全。在用户的使用过程中，需要经历身份审核、证书下载流程后，才能够使用签名业务。

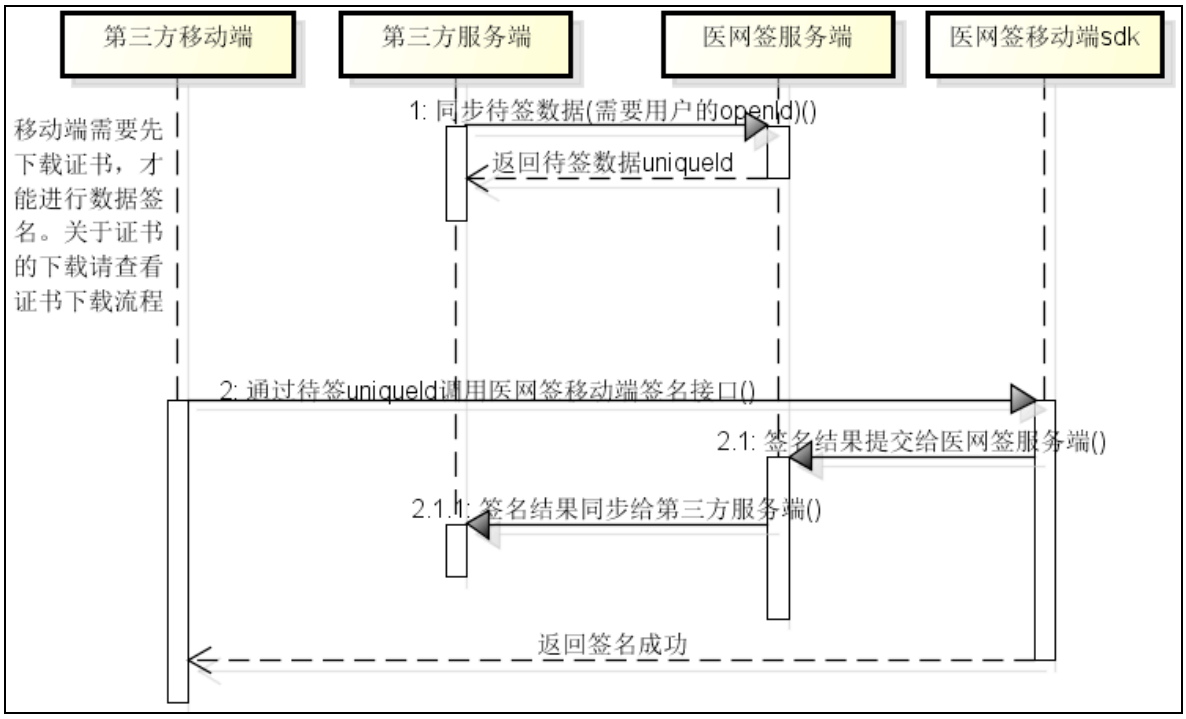
## 1.1 证书下载

用户需要通过第三方进行身份信息的同步，医网签将会审核该用户，审核成功后，用户需要下载证书，才能进行后续的各种签名业务，关于证书下载的流程如下图：



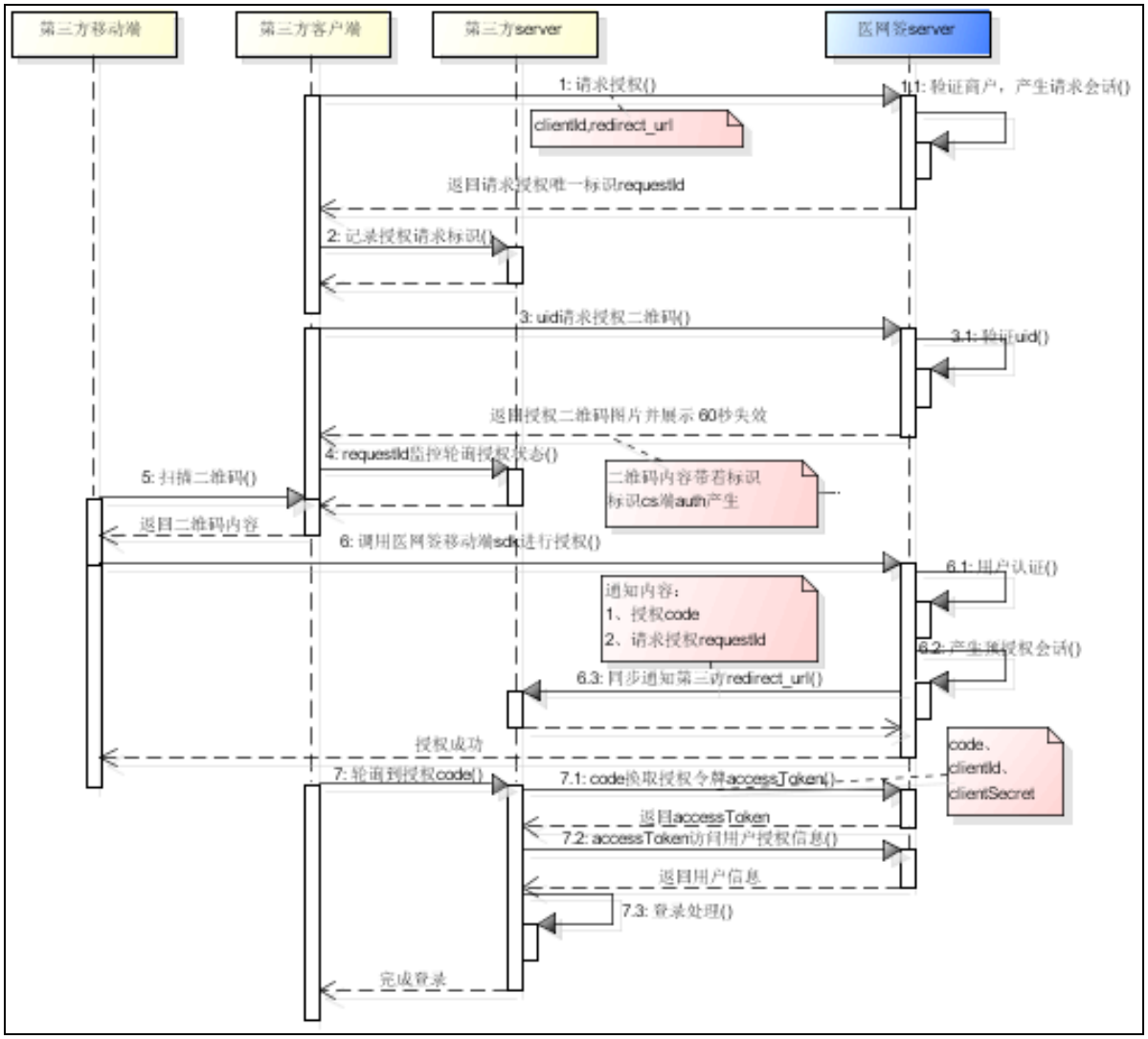

## 1.2 数据签名

用户下载医网签证书后，就可以进行数据签名，签名完成后医网签服务端会将签名结果同步给第三方服务器端，**移动端的 sdk 模块只负责对数据进行签名，不会把签名结果直接返回给调用的第三方移动端**。基础的签名流程如下图：




## 1.3 PC 二维码 OAuth 认证登录

PC 二维码 OAuth 登录应用场景是第三方客户端通过医网签平台对证书用户进行授权登录第三方业务系统。




## 1.4 PC 二维码签名

Pc 二维码签名应用场景为第三方同步待签数据到医网签后，通过医网签返回的二维码生成 url 展示待签名二维码，第三方移动客户端可以通过扫描二维码进行数据签名。

# 2 开发环境要求

## 2.1 Android 集成环境要求

**Android SDK:** android 4.0 以上版本
Java JDK: 1.6 以上版本
Android 系统: 4.0 以上版本
开发工具: Android studio/Eclipse

# 3 Android 集成流程

## 3.1 集成前配置工作

### 3.1.1 Android studio 开发工具下集成

1 导入关键 aar 包 doctor-sdk.aar 到 libs 目录中
2 在 module 中的 build.gradle 文件的 dependencies 节点中添加以下配置

```
compile(name: 'doctor-sdk', ext: 'aar')
compile 'com.google.code.gson:gson:2.3.1'
```

3 将 so 库导入到 libs 目录中(so 的导入可以参考网上配置)，在 module 中的 build.gradle 中 android 节点下增加一下内容指定 so 的存放目录。关于 so 库引用的问题可以参考网上资源
(http://blog.csdn.net/ouyang_peng/article/details/51168072)

```
sourceSets {
    main {
        jniLibs.srcDirs = ['libs']
    }
}
```

4 配置医网签服务地址：**在使用医网签 sdk 异步接口前，请务必通过代码配置对应的环境地址。**其中 EnvType.PUBLIC 为医网签**正式**环境，EnvType.INTEGRATE 为**集成**环境。用户首先在集成环境下进行集成，集成完成后，再移动端和服务端同时切换到医网签的正式环境，并要求医网签运营人员配置厂商唯一标识 clientId 即可。

```
BJCASDK.getInstance().setServerUrl(EnvType.INTEGRATE);
```

### 3.1.2 接口中重要参数说明

| | |
|---|---|
| uniqueId | 来源：第三方同步待签数据到医网签服务端产生<br>应用场景：第三方移动端进行数据签名业务 |
| clientId | 来源：医网签为第三方厂商提供的唯一标识<br>应用场景：所有和医网签业务相关的场景 |
| openId | 来源：第三方同步用户信息到医网签服务端返回的用户唯一标识<br>应用场景：待签名数据的提交等场景 |

### 3.1.3 Android 混淆处理

在集成过程中，不对医网签内部文件混淆

```
#将以下的代码段添加到  混淆配置文件中
#保持 Serializable 不被混淆
-keepnames class * implements java.io.Serializable
# 不混淆 gson 代码
-keep class com.google.gson.** {*;}
# 不混淆医网签所有文件
-dontwarn   cn.org.bjca.**
-keep class cn.org.bjca.**{ *;}
-keep interface cn.org.bjca.**{ *;}
```

## 3.2 接口列表

### 3.2.1 设置医网签 sdk 地址

| | |
|---|---|
| 功能 | 配置医网签 sdk 的环境类型（默认为正式环境），**环境如果发生变更，必须完全重启应用才能生效** |
| 函数定义 | `void setServerUrl(Envtype envType)` |
| 参数 | `envType: [IN]` 医网签环境类型（集成环境 `EnvType.INTEGRATE`、正式环境 `EnvType.PUBLIC`） |
| 返回值 | 当前配置的医网签环境的地址前缀 |

❑ 调用过程:

```
BJCASDK.getInstance().setServerUrl(EnvType.INTEGRATE);
```

### 3.2.2 启动首页

| | |
|---|---|
| 功能 | 打开首页 |
| 函数定义 | `void startDoctor(Context context,String clientId)` |
| 参数 | `context: [IN]` 接口调用的 `activity` 上下文<br>`clientId: [IN]` 第三方厂商标识 |
| 返回值 | 无 |

❑ 调用过程:

```
BJCASDK.getInstance().startDoctor(context, clientId);
```

### 3.2.3 页面跳转

| 功能 | 进行页面的选择跳转（可以结合判断证书、签章是否存在接口进行使用） |
|---|---|
| 函数定义 | void startUrl(Context context,String clientId, String page) |
| 参数 | context: [IN] 接口调用 activity 上下文。<br>clientId: [IN] 第三方厂商标识<br>page: [IN] 跳转页面标识：0 首页 1 下载证书 2 找回证书 3 设置签章 4 更新证书 |
| 返回值 | 无 |
| 调用过程：<br>BJCASDK.*getInstance*().startUrl(MainActivity.this, clientId, page); | |

### 3.2.4 签名—单次

| 功能 | 签名待签数据 |
|---|---|
| 函数定义 | int signRecipe(Context context, String clientId, String uniqueId) |
| 参数 | context :[IN] 接口调用 activity 上下文。<br>clientId:[IN] 厂商标识 ID<br>uniqueIds:[IN] 待签数据同步到医网签返回的 uniqueId 的列表 |
| 返回值 | int 调用返回码，详见附录 1 |

调用过程：

```
BJCASDK.getInstance().signRecipe(context, clientId, uniqueId);
```

方法回调：

```
@Override
protected void onActivityResult(int requestCode, int resultCode,
Intent data) {
    if (requestCode == ConstantParams.ACTIVITY_SIGN_DATA) {
        String result =
data.getStringExtra(ConstantParams.KEY_SIGN_BACK);
      }
}
```

1.签名返回示例：
{"status":"0",
"message":"success",
"clientId":"2015112716143758",
"stamp":"xxxxxxxxxxxx",
"stampPic":"xxxxxxxxxxxxxxxxx"
"uniqueIds":["7d8d1b7fade942a4954ab8fe7ffa851a"]}
**2.签名返回参数说明：**
status:  //状态码，当为“0”的时候业务成功
message:签名结果说明
clientId://厂商标识
stamp:  //签章值（建议以医网签服务端给厂商服务端返回的签章值为标准）
stampPic://图章(应用厂商可以用于展示，但建议以服务端验章返回的图片为标准)
uniqueIds: //待签数据的医网签唯一标识 uniqueId 列表

## 3.2.5 签名—批量

| 功能 | 医师对待签数据进行批量签名 |
|---|---|
| 函数定义 | int signBatch(Activity activity,String clientId,List<String> uniqueIdList); |
| 参数 | context :[IN] 接口调用 activity 上下文。<br>clientId:[IN] 厂商标识 ID<br>uniqueIds:[IN] 待签数据同步到医网签返回的 uniqueId 的列表 |
| 返回值 | int 调用返回码，详见附录 1 |

调用过程：
```
int result=BJCASDK.getInstance().signBatch(this,clientId,listUniqueId);
```
方法回调：
```
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    if (requestCode == ConstantParams.ACTIVITY_SIGN_DATA) {
        String result = data.getStringExtra(ConstantParams.KEY_SIGN_BACK);
      }
}
```

1.签名返回示例：
{"clientId":"2015112716143758","message":"success","status":"0",
"stamp":"xxxxxxxxxxxx",
"stampPic":"xxxxxxxxxxxxxxxxx"

```
"uniqueIds":["7d8d1b7fade942a4954ab8fe7ffa851a"]}
2.签名返回参数说明：
status: // 状态码，当为”0”的时候业务成功
message: //签名结果说明
clientId: //厂商标识
stamp:    //签章值（建议以医网签服务端给厂商服务端返回的签章值为标准）
stampPic: //图章（应用厂商可以用于展示，但建议以服务端验章返回的图片为标准）
uniqueIds: //待签数据的医网签唯一标识 uniqueId 列表
```

## 3.2.6 医网签二维码处理（推荐使用）

| | |
|---|---|
| 功能 | 移动端设备扫描医网签业务二维码进行处理(二维码的生成必须是由第三方服务端调用医网签服务端同步待签数据产生的，移动端 sdk 仅接收二维码的值，关于二维码的解析由第三方移动端自行处理)，当前仅**支持二维码签名和二维码 OAuth 认证** |
| 函数定义 | int qrDispose(Context context, String clientId, String jsonData) |
| 参数 | context :[IN] 接口调用 activity 上下文。<br>clientId:[IN] 厂商标识 ID<br>jsonData:[IN] 二维码扫描后数据 |
| 返回值 | int 调用返回码，详见附录 1 |

调用过程：

```
BJCASDK.getInstance().qrDispose(context, clientId, jsonData);
```

方法回调：

```
@Override
protected void onActivityResult(int requestCode, int resultCode,
Intent data) {
switch (requestCode) {
    //二维码签名返回
    case ConstantParams.ACTIVITY_QR_SIGN:{
 String result = data.getStringExtra(ConstantParams.KEY_SIGN_BACK);
    break;
    }
    //OAuth 认证登录返回
    case ConstantParams.ACTIVITY_QR_OAUTH:
 String result = data.getStringExtra(ConstantParams.KEY_OAUTH_BACK);
     break;
    default:
      break;
```

```
}
```

1.二维码操作返回结果集：

```
{"status":"0",
"message":"success"
}
```

**2.签名返回参数说明：**

```
status:  //状态码，当为"0"的时候业务成功
message: //二维码业务结果说明
```

## 3.2.7 PC 二维码签名

| 功能 | 移动端设备扫描 PC 二维码进行数据签名(二维码的生成必须是由第三方服务端调用医网签服务端同步待签数据产生的，移动端 sdk 仅接收二维码的值，关于二维码的解析由第三方移动端自行处理) |
|---|---|
| 函数定义 | int qrSignRecipe(Context context, String clientId, String jsonData) |
| 参数 | context :[IN] 接口调用 activity 上下文。<br>clientId:[IN] 厂商标识 ID<br>jsonData:[IN] 二维码扫描后数据 |
| 返回值 | int 调用返回码，详见附录 1 |

调用过程：

```
BJCASDK.getInstance().qrSignRecipe(context, clientId, jsonData);
```

方法回调：

```
@Override
protected void onActivityResult(int requestCode, int resultCode,
Intent data) {
    if (requestCode == ConstantParams.ACTIVITY_QR_SIGN) {
        String                          result                          =
data.getStringExtra(ConstantParams.KEY_SIGN_BACK);
      }
}
```

**1.签名返回示例：**

```
{"status":"0",
"message":"success"
}
```

**2.签名返回参数说明：**

```
status:  //状态码，当为"0"的时候业务成功
message: //签名结果说明
```

### 3.2.8 PC 二维码 OAuth 登录

| | |
|---|---|
| 功能 | 通过移动设备对 PC 的二维码扫描，完成身份认证登录(pc 的 OAuth 认证二维码由第三方服务端调用医网签服务端获取。扫码模块由第三方自行处理，本接口不做扫码处理，只接收扫码得到的字符串进行处理) |
| 函数定义 | int qrOauth(Activity activity, String clientId, String qrText); |
| 参数 | activity: [IN] 接口调用 activity 上下文。<br>clientId: [IN] 第三方厂商的 clientId<br>qrText: [IN] 二维码的值 |
| 返回值 | int 调用返回码，详见附录 1 |

**1.调用过程：**

```
BJCASDK.getInstance().qrOauth(activity, clientId, qrText);
```

**2.方法回调：**

```
@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    if (requestCode == ConstantParams.ACTIVITY_QR_OAUTH) {
        String result=data.getStringExtra(ConstantParams.KEY_OAUTH_BACK);
       }
}
```

**1.签名返回示例：**

```
{"status":"0",

"message":"success"

}
```

**2.签名返回参数说明：**

```
status:   //状态码，当为”0”的时候业务成功
message:  //签名结果说明
```

### 3.2.9 判断证书是否存在

| | |
|---|---|
| 功能 | 判断本地是否下载证书 |
| 函数定义 | boolean existsCert(Context context) |
| 参数 | context: [IN] 接口调用 activity 上下文 |
| 返回值 | true 本地存在证书 false 本地不存在证书 |

调用过程：

```
boolean existsCert = BJCASDK.getInstance().existsCert(context)
```

### 3.2.10 判断用户是否设置个人签章

| | |
|---|---|
| 功能 | 判断用户是否已经设置了个人签章（签名图章） |
| 函数定义 | boolean existsStamp(Context context) |
| 参数 | context：[IN] 接口调用 activity 上下文 |
| 返回值 | true 用户已经设置了个人签章 false 没有设置 |
| 调用过程：<br>boolean existsStamp = BJCASDK.getInstance().existsStamp(context); | |

### 3.2.11 判断用户是否处于免密签名状态

| | |
|---|---|
| 功能 | 判断用户是否处于免密签名状态 |
| 函数定义 | boolean isPinExempt(Context context) |
| 参数 | context：[IN] 接口调用 activity 上下文 |
| 返回值 | true-用户当前处于免密签名状态 false-其他 |
| 调用过程：<br>boolean isPinExempt= BJCASDK.*getInstance*().isPinExempt(context) | |

### 3.2.12 获取用户唯一标识 openId

| | |
|---|---|
| 功能 | 获取用户的 openId |
| 函数定义 | String getOpenId(Context context) |
| 参数 | context：[IN] 接口调用 activity 上下文 |
| 返回值 | 用户在医网签的唯一标识 openId,用户证书不存在时返回 null |
| 调用过程：<br>String openId= BJCASDK.*getInstance*().getOpenId(context) | |

### ~~3.2.13 获取证书用户信息(在 2.1.1 版本不再支持)~~

### 3.2.14 获取用户信息

| | |
|---|---|
| 功能 | 获取用户信息。通过此接口能获取证书有效期等相关信息，可以在证书过期前一段时间提醒用户进行证书更新操作。（证书存在时才能调用此接口） |
| 函数定义 | getUserInfo(Context context) |
| 参数 | context：[IN] 接口调用 activity 上下文 |
| 返回值 | UserBean 对象 |

**1.调用过程：**

```
BJCASDK.getInstance().getUserInfo(context,clientId,new
OperateListener(){
  public void callback(String message){
     //用户信息返回
  }
};
```

**2.String 返回说明：**

```
private String status; //状态码，为“0”时成功
private String message;//返回说明
private String openId; //用户唯一标识
private String userName; //用户名
private String startTime; //证书颁发时间
private String endTime; //证书有效时间
private String nowTime; //医网签服务器当前时间
private String certUpdateTipDay; //证书更新提前提醒天数。假设为 30 天，即证书到期前 30 天可以进行证书更新操作
private String existsStamp;//当前用户是否已经设置个人签章
```

## 3.2.15 清除本地证书

| 功能 | 清除本地证书 |
|---|---|
| 函数定义 | boolean clearCert(Context context) |
| 参数 | context：[IN] 接口调用 activity 上下文 |
| 返回值 | 证书唯一标识 openId，没有证书则返回空 |
| 调用过程：<br>String openId= BJCASDK.getInstance().clearCert(context) | |

## 3.2.16 清除密码

| 功能 | 清除用户在签名过程中保存的密码(**用户签名完成后，会提示用户记住密码**) |
|---|---|
| 函数定义 | boolean clearPin(Context context) |
| 参数 | context：[IN] 接口调用 activity 上下文 |
| 返回值 | true—成功清除签名密码　false—清除签名密码出错 |
| 调用过程：<br>boolean success= BJCASDK.getInstance().clearPin(context) | |

# 4 附录 1

## 4.1 错误码说明（同步）

| 返回码 | 类型 | 说明 |
| --- | --- | --- |
| 0 | int | 成功 |
| 1001 | int | 证书不存在 |
| 1002 | int | 签章不存在 |
| 1003 | int | 参数为空 |
| 1004 | int | 程序运行缺少权限 |
| 1006 | int | 批量签名数量错误--不能大于 100 条 |
| 2002 | int | 当前二维码业务医网签无法处理 |

## 4.2 错误码说明（签名过程在 onActivityResult 中体现）

| 错误码 | 类型 | 错误描述 |
| --- | --- | --- |
| 0 | String | 操作成功 |
| 2001 | String | 网络异常 |
| 2002 | String | 二维码解析错误 |
| 3000 | String | 用户取消操作 |
| 003x070 | String | 当前待签数据状态异常，请联系管理员查询原因 |
| 004x001 | String | 用户证书异常（用户需要重新下载或找回） |
| 004x009 | String | 用户证书被停用 |
| 004x030 | String | 用户证书已过期（需要引导用户进行证书更新） |
| 002x033 | String | 用户证书已被注销 |
| 002x039 | String | 当前设备不是用户绑定的设备（用户可能在别的设备上进行了找回证书操作，可以引导用户进行证书找回） |