阿里云
实人认证

活体人脸验证

文档版本：20190816

法律声明

阿里云提醒您在阅读或使用本文档之前仔细阅读、充分理解本法律声明各条款的内容。如果您阅读或使用本文档，您的阅读或使用行为将被视为对本声明全部内容的认可。

1. 您应当通过阿里云网站或阿里云提供的其他授权通道下载、获取本文档，且仅能用于自身的合法合规的业务活动。本文档的内容视为阿里云的保密信息，您应当严格遵守保密义务；未经阿里云事先书面同意，您不得向任何第三方披露本手册内容或提供给任何第三方使用。
2. 未经阿里云事先书面许可，任何单位、公司或个人不得擅自摘抄、翻译、复制本文档内容的部分或全部，不得以任何方式或途径进行传播和宣传。
3. 由于产品版本升级、调整或其他原因，本文档内容有可能变更。阿里云保留在没有任何通知或者提示下对本文档的内容进行修改的权利，并在阿里云授权通道中不时发布更新后的用户文档。您应当实时关注用户文档的版本变更并通过阿里云授权渠道下载、获取最新版的用户文档。
4. 本文档仅作为用户使用阿里云产品及服务的参考性指引，阿里云以产品及服务的”现状“、“有缺陷”和“当前功能”的状态提供本文档。阿里云在现有技术的基础上尽最大努力提供相应的介绍及操作指引，但阿里云在此明确声明对本文档内容的准确性、完整性、适用性、可靠性等不作任何明示或暗示的保证。任何单位、公司或个人因为下载、使用或信赖本文档而发生任何差错或经济损失的，阿里云不承担任何法律责任。在任何情况下，阿里云均不对任何间接性、后果性、惩戒性、偶然性、特殊性或刑罚性的损害，包括用户使用或信赖本文档而遭受的利润损失，承担责任（即使阿里云已被告知该等损失的可能性）。
5. 阿里云网站上所有内容，包括但不限于著作、产品、图片、档案、资讯、资料、网站架构、网站画面的安排、网页设计，均由阿里云和/或其关联公司依法拥有其知识产权，包括但不限于商标权、专利权、著作权、商业秘密等。非经阿里云和/或其关联公司书面同意，任何人不得擅自使用、修改、复制、公开传播、改变、散布、发行或公开发表阿里云网站、产品程序或内容。此外，未经阿里云事先书面同意，任何人不得为了任何营销、广告、促销或其他目的使用、公布或复制阿里云的名称（包括但不限于单独为或以组合形式包含”阿里云”、Aliyun”、“万网”等阿里云和/或其关联公司品牌，上述品牌的附属标志及图案或任何类似公司名称、商号、商标、产品或服务名称、域名、图案标示、标志、标识或通过特定描述使第三方能够识别阿里云和/或其关联公司）。
6. 如若发现本文档存在任何错误，请与阿里云取得直接联系。

通用约定

格式	说明	样例
	该类警示信息将导致系统重大变更甚至故障，或者导致人身伤害等结果。	 **禁止:** 重置操作将丢失用户配置数据。
	该类警示信息可能导致系统重大变更甚至故障，或者导致人身伤害等结果。	 **警告:** 重启操作将导致业务中断，恢复业务所需时间约10分钟。
	用于补充说明、最佳实践、窍门等，不是用户必须了解的内容。	 **说明:** 您也可以通过按Ctrl + A选中全部文件。
>	多级菜单递进。	设置 > 网络 > 设置网络类型
粗体	表示按键、菜单、页面名称等UI元素。	单击 **确定**。
`courier` 字体	命令。	执行 `cd /d C:/windows` 命令，进入Windows系统文件夹。
##	表示参数、变量。	`bae log list --instanceid` *`Instance_ID`*
[]或者[a|b]	表示可选项，至多选择一个。	`ipconfig` *`[-all|-t]`*
{}或者{a|b}	表示必选项，至多选择一个。	`swich` *`{stand | slave}`*

目录

1 活体人脸验证接入流程

该接入流程文档适用于活体人脸验证服务。

单击查看老版本接入文档说明。

准备工作

1. 前往阿里云官网注册账号。如果已有注册账号，请跳过此步骤。
2. 对该账号进行#unique_5。如果已经是企业账号，请跳过此步骤。
3. 打开云盾实人认证产品页面，单击立即开通，开通实人认证服务。
4. 在接入之前，可以根据业务上的需求，先了解#unique_6/unique_6_Connect_42_section_13r_whe_p8p，选择可以满足业务需求的认证方案。
5. 登录阿里云实人认证控制台，在接入及设置页面，创建场景并进行相应的流程配置，具体操作见#unique_7/unique_7_Connect_42_section_gbr_hh7_02f。
6. 根据#unique_8，了解客户端和服务端的交互流程，及各自需要做的事情。

完成上述操作之后，接入方客户端和服务端开发可以介入，进行相应的集成开发和联调测试。

客户端集成无线SDK

1. #unique_9，用于在手机APP中集成。
2. 根据应用类型完成对应的集成操作。
   - Android SDK集成
   - iOS SDK集成
3. 集成客户端SDK之后，如果想要对活体UI有些定制化修改，可以参见SDK UI自定义说明进行调整。

服务端接入

1. 阿里云API的调用需要使用AccessKey来校验请求数据的安全性，接入方需要在AccessKey管理页面获取AccessKey。但因为阿里云账号的 AccessKey 具有所有云产品API的访问权限，一旦泄露将导致极大的安全风险，所以强烈建议您根据最小权限原则创建并使用RAM子用户来进行API访问和控制台操作，具体参见RAM子账号接入。
2. 获取到AccessKey之后，参见API调用方式了解请求的结构和公共参数。

3. 服务端API接口概览。

API	描述
#unique_10（DescribeVerifyToken）	获取认证token。适用于SDK+服务端接入、H5+服务端接入的认证方案，在发起认证时调用。
#unique_11（DescribeVerifyResult）	查询认证结果。适用于SDK+服务端接入、H5+服务端接入的认证方案，认证后查询认证结果。

4. 实际接入具体的服务端API接口时，推荐使用我们的服务端SDK开发包。服务端SDK开发包中有对应的示例代码，便于接入方接入和调试。
5. 阿里云也提供了一个OpenAPI调用工具，可供接入方调试实人认证服务端API。

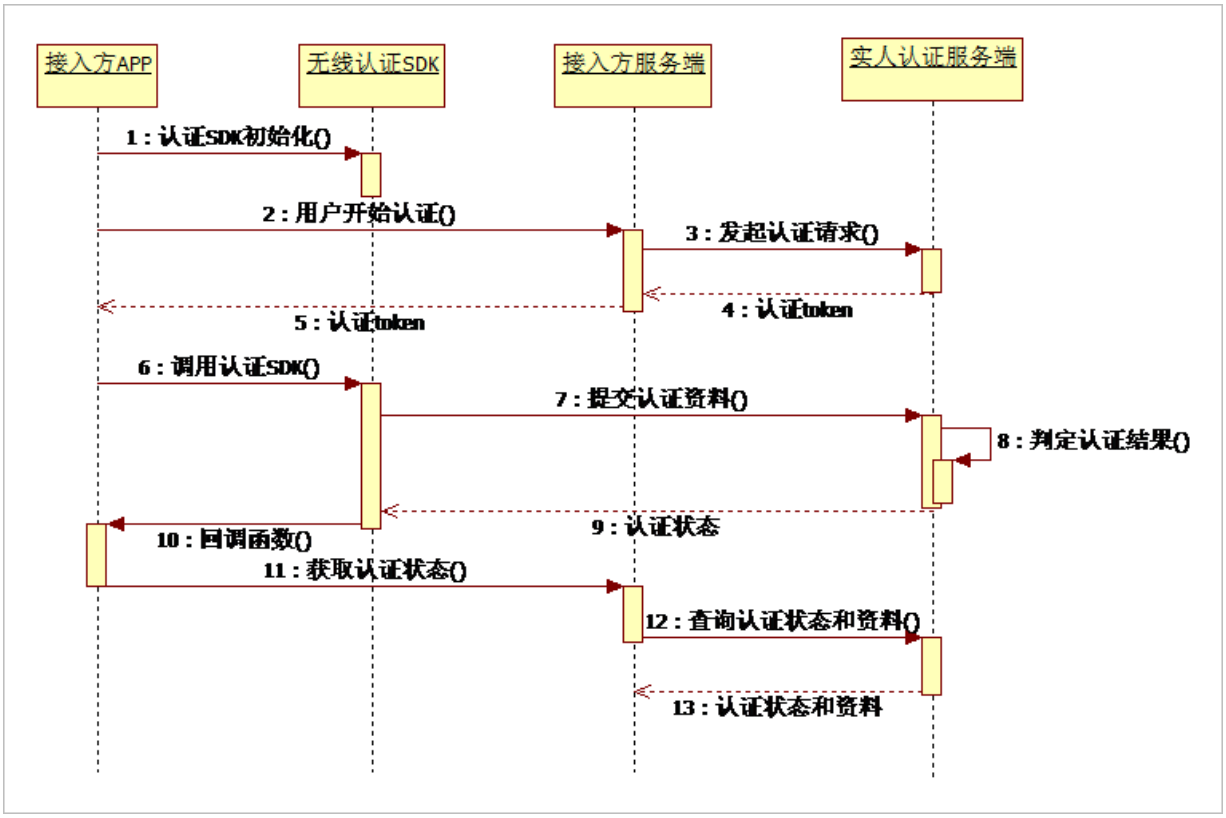
2 集成指南

2.1 接入时序图

本章节内容适用于活体人脸验证产品的接入时序参考。

单击查看老版本接入文档说明。

无线SDK+服务端接入



时序图说明：

- Step1：接入方在移动APP中集成接入了无线认证SDK，并进行SDK初始化。
- Step2：用户进入接入方APP的认证业务入口，准备开始认证。
- Step3~Step5：接入方服务端调用发起认证请求接口，从实人认证服务端获取token；同时，根据不同的认证方案，可以通过该接口提前传入一些材料。取到token后返回给接入方APP端。
- Step6：接入方APP调起认证SDK，用户进入认证操作流程，根据不同的认证方案，需要用户做不同的操作比如做动作活体、拍摄身份证正反面等。
- Step7：用户操作完成后，由无线SDK将本次用户操作提交到实人认证服务端。
- Step8：实人认证服务端根据综合决策逻辑，进行认证结果判定。

- Step9~Step10：实人认证服务端将认证结果回调给无线SDK，并由无线SDK将结果回调告知接入方APP。
- Step11~Step13：接入方APP接收到认证结果回调后，此时接入方服务端就可以调用获取认证状态的接口，从实人认证服务端来获取认证状态和认证资料。（客户端回调的认证状态存在一定可能性会被恶意篡改，建议由接入方服务端调接口来实人认证服务端获取最终可信的状态，并据此做业务上的决策）。

适用的认证方案：FVBioOnly、FDBioOnly

无线SDK端接入：

- #unique_9
- Android SDK集成、iOS SDK集成

服务端接入：#unique_15

API接口：发起认证请求（DescribeVerifyToken）、查询认证状态和资料（VerifyMaterial）

2.2 无线SDK接入

2.2.1 下载无线SDK

本文介绍了下载活体人脸验证无线SDK的具体步骤。

操作步骤

1. 使用阿里云账号或具有AliyunYundunCloudAuthFullAccess授权 的RAM子用户，登录 实人认证管理控制台。

   说明:
   关于创建RAM子用户、给RAM用户授权、RAM用户登录控制台的方法，请参见RAM子账号接入。

2. 在左侧导航栏，单击接入及设置。

3. 打开活体人脸验证页签，选择要操作的认证场景，并单击获取认证SDK。

4. 在认证SDK生成对话框中，选择应用类型（iOS/Android），并单击上传应用。

5. 从本地选择需要集成认证SDK的应用，并上传其ipa文件（iOS应用）或apk文件（Android应用）。

> **说明:**
>
> - 无线认证SDK是绑定应用的，因此需要先上传您的应用以便系统为您生成该应用专属的SDK。如果应用版本迭代过程中，包名和签名（或bundleid）未变化，则无需重新生成SDK，如果包名或签名有变化，则需要重新上传应用，生成SDK再集成使用。
> - 在流程配置中，修改了引导页、用户授权声明页、结果页，不需要重新下载更新认证SDK。

6. 应用上传完成后，单击下载认证SDK完成下载。

7. 参见以下文档完成无线认证SDK的集成：

   - Android SDK集成
   - iOS SDK集成

2.2.2 Android集成

下载无线认证 SDK 后，您可参考以下步骤将认证 SDK 集成到您的 Android 应用中。

背景信息

单击查看老版本接入文档说明。

Android SDK 与包名（package name）+签名（keystore）绑定，修改package name或keystore都需要在管理控制台上重新下载SDK，debug和release包在使用不同签名时不能混用。

说明:

若您需要用到 V2 方式的签名，打包时请同时勾选 V1、V2 签名（如果只选择V2签名，apk将无法在Android 7.0以下安装）。

在工程中导入SDK

说明:

对于已经接入过实人认证无线SDK的接入方（rpsdk版本号是2.1.x.x），在控制台下载到新版本的无线SDK后（rpsdk 版本号3.1.x.x），需要将新下载SDK包中的所有的文件覆盖到原有开发包，并按下述步骤引入到应用工程中。

解压无线认证SDK包中的client.zip文件，将以下Android依赖包引入到您的应用工程中：

- oss-android-sdk-x.x.x.x.aar
- FaceLivenessOpen-x.x.x.x.aar
- NoCaptchaSDK-external-release.aar（通过解压`Android.NoCaptchaSDK.xxx.tar`获得该依赖包）
- SecurityBodySDK-external-release.aar（通过解压`Android.SecurityBodySDK.xxx.tar`获得该依赖包）
- SecurityGuardSDK-external-release.aar（通过解压`Android.SecurityGuardSDK.xxx.tar`获得该依赖包）
- Okhttp.jar Okio.jar
- logging-interceptor.jar
- Rpsdk.aar

- Windvane-mini.jar

例如，如果您使用的AndroidStudio，可参考以下步骤：

1. 设定依赖包所在的路径。

```
apply plugin: 'com.android.application'
 repositories {
     flatDir {
         dirs '../libs'
     }
 }
```

2. 设定引入的本地库所在路径，将需要引入的依赖包都放在*../libs*目录，包含所有需要的库。

3. 在*gradle*文件中引入以下需要的库依赖。

```
 compile fileTree(dir: '../libs', include: ['*.jar'])
 compile (name:'aliyun-oss-sdk-android-2.9.2',ext:'aar')
 compile (name:'FaceLivenessOpen-2.1.6.10',ext:'aar')
 compile (name:'rpsdk-2.4.0.3',ext:'aar')
 compile (name:'SecurityGuardSDK-external-release-5.4.94',ext:'aar')
 compile (name:'SecurityBodySDK-external-release-5.4.66',ext:'aar')
 compile (name:'NoCaptchaSDK-external-release-5.4.26',ext:'aar')
```

关于签名图片

- 解压已下载的无线认证SDK包，获得*yw_1222_*.jpg*签名图片文件，该文件用于无线认证SDK的授权。
- 把该图片文件导入到工程中*res\drawable*目录，如果没有这个文件夹，请先在工程中创建，否则将无法正常工作。
- 如果在安卓工程打包时启用了shrinkResources true，还需要在*keep.xml*文件中添加以下内容：

```
<resources xmlns:tools="http://schemas.android.com/tools" tools:
keep="@drawable/yw_1222_*" />
```

关于 CPU 类型

无线认证 SDK 目前支持 armeabi、armeabi-v7a、arm64-v8a，请接入方按需在*build.gradle*中增加abifilters配置。例如，接入方仅需要支持其中 armeabi 和 arm64-v8a，则配置如下：

```
defaultConfig {
    ndk {
        abiFilters "armeabi", "arm64-v8a"
    }
```

```
}
```

关于权限

无线认证SDK的使用需要添加以下权限：

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

关于应用主题配置

```
<item name="android:windowIsTranslucent">false</item>
```

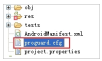
添加ProGuard 配置

如果您的应用使用了ProGuard 进行代码混淆，为了保证实人认证服务需要的一些类不被混淆，需要在 ProGuard 配置文件中添加相关指令。

1. 根据您接入方式的情况，判断是否使用了 ProGuard 进行代码混淆。

   - Eclipse

     如果在 project.properties中指定了ProGuard配置（例如，在`project.properties`中包含`proguard.config=proguard.cfg`语句），则表明已使用 ProGuard 进行代码混淆，混淆配置在 *proguard.cfg* 文件中。

   - Android Studio

     如果在*build.gradle* 中配置了 proguardFiles，并且启用了 minifyEnabled 方法，则表明已使用 *proguard-rules.pro* 这个配置文件进行代码混淆。例如：

     ```
     buildTypes {
       release {
           minifyEnabled true
           proguardFiles getDefaultProguardFile('proguard-android.txt
     '),     'proguard-rules.pro'
       }
     }
     ```

2. 在相应的ProGuard 配置文件中添加以下配置信息，保证实人认证服务需要的类不被混淆。

```
-keep class com.taobao.securityjni.**{*;}
 -keep class com.taobao.wireless.security.**{*;}
 -keep class com.ut.secbody.**{*;}
 -keep class com.taobao.dp.**{*;}
 -keep class com.alibaba.wireless.security.**{*;}
 -keep class com.alibaba.security.rp.**{*;}
 -keep class com.alibaba.sdk.android.**{*;}
```

```
 -keep class com.alibaba.security.biometrics.**{*;}
 -keep class android.taobao.windvane.**{*;}
```

使用SDK

1. 初始化。

   一般在应用启动时进行。

```
import com.alibaba.security.rp.RPSDK;
 RPSDK.initialize(appContext);
```

2. 进入认证。

```
RPSDK.start(verifyToken, ParametersActivity.this,
     new RPSDK.RPCompletedListener() {
     @Override
     public void onAuditResult(RPSDK.AUDIT audit, String code) {
         Toast.makeText(ParametersActivity.this, audit + "",
Toast.LENGTH_SHORT).show();
         if(audit == RPSDK.AUDIT.AUDIT_PASS) { //认证通过
         }
         else if(audit ==
RPSDK.AUDIT.AUDIT_FAIL) { //认证不通过
         }
         else if(audit == RPSDK.AUDIT.AUDIT_NOT) { //未认证，通常是用户
主动退出或者姓名身份证号实名校验不匹配等原因，导致未完成认证流程
         }
     }
 });
```

说明:

- verifyToken参数由接入方的服务端调用实人认证服务的DescribeVerifyToken接口获得。
- 在*RPSDK.start*接口调用的回调会返回用户认证的各种状态，onAuditResult函数的的audit和code参数取值，可参见下表说明。

audit	code	code释义
RPSDK.AUDIT. AUDIT_PASS	1	认证通过
RPSDK.AUDIT. AUDIT_FAIL	取值3~12	表示认证不通过，具体的不通过原因可以查看服务端的查询认证结果（DescribeVerifyResult）接口文档中认证状态的表格说明
RPSDK.AUDIT. AUDIT_NOT	-1	未完成认证，原因是：用户在认证过程中，主动退出

audit	code	code释义
RPSDK.AUDIT.AUDIT_NOT	3001	未完成认证，原因是：认证token无效或已过期
RPSDK.AUDIT.AUDIT_NOT	3101	未完成认证，原因是：用户姓名身份证实名校验不匹配
RPSDK.AUDIT.AUDIT_NOT	3102	未完成认证，原因是：实名校验身份证号不存在
RPSDK.AUDIT.AUDIT_NOT	3103	未完成认证，原因是：实名校验身份证号不合法
RPSDK.AUDIT.AUDIT_NOT	3104	未完成认证，原因是：认证已通过，重复提交
RPSDK.AUDIT.AUDIT_NOT	3204	未完成认证，原因是：操作太频繁

常见问题

- 调起无线认证SDK，进入认证页面展示“UNKNOWN_ERROR”/“网络异常，请检查网络”。

  处理建议：查看logcat。

  - 若SG ERROR= 1208、1215、1224、1411
    - 如果您已经接入过实人认证无线SDK（rpsdk版本号是2.1.x.x），在控制台下载到新版本的无线SDK后（rpsdk 版本号3.1.x.x），需要将新下载SDK包中的所有的文件覆盖到原来开发包中，否则会报错。
    - 如果确认不存在上述情况，则说明当前开发包与在管理控制台下载 SDK 时上传包的包名（packagename）或签名（keystore）不一致。
      - 请在管理控制台重新上传当前开发包，用新下载 SDK 中的*yw_1222_*.jpg*签名图片文件替换开发包中原有的文件。
      - 若上述操作无法解决问题，确认签名图片是否正确放在*res\drawable*目录下，且且关闭了Instant Run调试模式。
  - 若SG ERROR= 1412、1225，说明图片不存在，请确保*res\drawable*目录下有正确的签名图片，且关闭了Instant Run调试模式。

  说明:

工程上，通常对 IDE 中直接运行（debug）和正式打包（release）会配置不同的签名（keystore），在 IDE 中直接编译运行的是 debug 包，其签名图片文件和 release 包是不同的。

- 项目中之前引入的组件与无线认证 SDK 中的组件有重复，例如 SecurityGuardSDK、oss-android-sdk 等。

  处理建议：若 SecurityGuardSDK 组件有重复，删除低版本，但保留两个版本的*yw_1222_*.jpg*签名图片文件；若 oss-android-sdk 等其他组件有重复，删除低版本即可。

- 项目中之前引入的SecurityEnvSDK 组件与无线认证 SDK 中的SGMain 组件报 duplicate symbol 冲突。

  处理建议：SecurityEnvSDK 组件是 SGMain 组件的早先版本，删除 SecurityEnvSDK，但保留两个组件的*yw_1222_*.jpg*签名图片文件。

- APP 采用了插件机制，集成无线认证 SDK要注意什么?

  处理建议：建议把无线认证 SDK 放到主 Bundle 下。

2.2.3 iOS集成

下载无线认证SDK后，您可参考以下步骤将认证SDK集成到您的iOS应用中。

背景信息

单击查看老版本接入文档说明。

说明:
iOS SDK与BundleID绑定，不同BundleID需要在管理控制台上重新下载SDK。

在工程中导入 SDK

说明:
对于已经接入过实人认证无线SDK的接入方（iOS SDK版本号<=2.1，可在*RPSDK.framework*文件夹下*info.plist*文件中查看CFRPSDKVersion），在控制台下载到新版本的无线SDK后（iOS SDK版本号>=2.2），需要将新下载SDK包中的所有的文件替换原来工程里的对应文件，否则会报错。

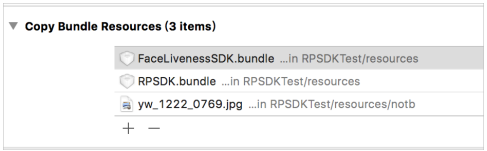
1. 解压无线认证SDK后，将以下iOS的依赖包引入到您的应用工程中：

   - RPSDK.framework
   - FaceLivenessOpen.framework
   - SecurityGuardSDK.framework
   - SGMain.framework
   - SGNoCaptcha.framework
   - SGSecurityBody.framework
   - AliyunOSSiOS.framework
   - WindVane.framework
   - WindVaneBasic.framework
   - WindVaneBridge.framework
   - WindVaneCore.framework
   - ZipArchive.framework
   - AliReachability.framework

2. 确认您的工程中已引入以下实人认证服务需要的系统依赖：

   - CoreMedia.framework
   - CoreMotion.framework
   - CoreTelephony.framework
   - AVFoundation.framework
   - ImageIO.framework
   - MobileCoreServices.framework
   - MediaPlayer.framework
   - CoreLocation.framework
   - AddressBook.framework
   - AddressBookUI.framework
   - SystemConfiguration.framework
   - AudioToolbox.framework
   - AssetsLibrary.framework
   - Libresolv.tbd
   - WebKit.framework
   - Libiconv.tbd
   - Libc++.tbd
   - Libz.tbd

3. 在您的工程资源中（Copy Bundle Resources），引入无线认证SDK 包中的*yw_1222_*.jpg*签名图片文件，以及*resource*目录下的*FaceLivenessSDK.bundle*和*RPSDK.bundle*文件。



4. 编译选项：在工程的Other Linker Flags选项中添加-ObjC。

使用SDK

1. 初始化。

   一般在应用启动时进行。

```
#import <RPSDK/RPSDK.h>
```

```
 [RPSDK initialize:RPSDKEnvOnline] //必须为RPSDKEnvOnline
```

2. 进入认证。

```
[RPSDK start:verifyToken rpCompleted:^(RPVerifyState verifyState,
NSString *code) {
     NSLog(@"verifyResult = %ld",(unsigned long)verifyState);
     if(verifyState == RPVerifyStatePass) { //认证通过。
     }
     else if(verifyState == RPVerifyStateFail) { //认证不通过。
     }
     else if(verifyState == RPVerifyStateNotVerify) { //未认证，用户主
动退出，或者姓名身份证号实名校验不匹配等原因，导致未完成认证流程。
     }

 }withVC:self.navigationController];
```

说明:

- verifyToken参数由接入方的服务端调用实人认证服务的DescribeVerifyToken接口获得。
- 在*RPSDK.start*接口调用的回调会返回用户认证的各种状态，具体信息体现在verifyState和code参数取值，详见下表说明。

verifyState	code	code释义
RPVerifyStatePass	1	认证通过
RPVerifyStateFail	取值3~12	表示认证不通过，具体的不通过原因可以查看服务端的查询认证结果（DescribeVerifyResult）接口文档中认证状态的表格说明
RPVerifyStateNotVerify	-1	未完成认证，原因是：用户在认证过程中，主动退出
RPVerifyStateNotVerify	3001	未完成认证，原因是：认证token无效或已过期
RPVerifyStateNotVerify	3101	未完成认证，原因是：用户姓名身份证实名校验不匹配
RPVerifyStateNotVerify	3102	未完成认证，原因是：实名校验身份证号不存在
RPVerifyStateNotVerify	3103	未完成认证，原因是：实名校验身份证号不合法
RPVerifyStateNotVerify	3104	未完成认证，原因是：认证已通过，重复提交

verifyState	code	code释义
RPVerifyStateNotVerify	3204	未完成认证，原因是：操作太频繁

常见问题

- 调起无线认证SDK，进入认证页面展示“网络异常，请检查网络”/“UNKNOWN_ERROR”/空白。

  处理建议：查看Xcode console。

  - 若SG ERROR= 1208、1215、1411
    - 如果您已经接入过实人认证无线SDK（iOS SDK版本号<=2.1，可在*RPSDK.framework*文件夹下*info.plist*文件中查看CFRPSDKVersion），在控制台下载到新版本的无线SDK后（iOS SDK版本号>=2.2），需要将新下载SDK包中的所有的文件替换原来开发包里的对应文件。
    - 如果确认不存在上述情况，则说明当前开发包与在管理控制台下载SDK时上传包的BundleID不一致。请在管理控制台重新上传当前开发包，用新下载SDK中的*yw_1222_*.jpg*签名图片文件替换开发包中原有的文件。
  - 若SG ERROR= 1412、1225，说明图片不存在。请确保SDK中的*yw_1222_*.jpg*签名图片正确地引入到了Copy Bundle Resource下。
- 项目中之前引入的组件与无线认证 SDK 中的组件有重复，例如SecurityGuardSDK、AliyunOSSiOS 等。

  处理建议：若SecurityGuardSDK组件有重复，删除低版本，但保留两个版本的*yw_1222_*.jpg*签名图片文件；若 AliyunOSSiOS等其他组件有重复，删除低版本即可。
- 项目中之前引入的SecurityEnvSDK组件与无线认证SDK中的SGMain组件报duplicate symbol冲突。

  处理建议：SecurityEnvSDK组件是SGMain组件的早先版本，删除SecurityEnvSDK，但保留两个组件的*yw_1222_*.jpg*签名图片文件。

2.2.4 SDK UI自定义说明

SDK中的活体环节UI，我们支持了自定义能力，具体操作方法请参见本文说明。使用SDK UI自定义前，请确保您已经在实人认证控制台下载安卓或iOS的无线端SDK。

活体环节自定图片

操作步骤

- Android

  1. 从实人认证控制台下载安卓无线SDK，如果本地之前已经下载过无线SDK，则用新下载SDK里的最新版*FaceLivenessOpen*替换本地旧版的*FaceLivenessOpen*。
  2. 将需要替换的图片放在*assets/facetheme/*下。

- iOS

  1. 从实人认证控制台下载iOS无线SDK，如果本地之前已经下载过无线SDK，则用新下载SDK里的*FaceLivenessOpen.framework*替换本地旧版的*FaceLivenessOpen.framework*。
  2. 将需要替换的图片放在*FaceTheme.bundle*的*images*目录下。
  3. 将*FaceTheme.bundle*加入项目中。

原始图片说明

原始图片可在*FaceTheme.bundle*中的*images*目录中找到，说明如下：

- 屏幕下方按钮：*face_nav_button.png*
- 屏幕顶端关闭按钮：*face_top_back.png*
- 屏幕顶端声音按钮：
  - *face_top_sound_on.png*（声音打开）
  - *face_top_sound_off.png*（声音关闭）
- 中间圆内圈-开始时底图：*face_circle_inner_bg.png*
- 中间圆内圈-检测到人脸时旋转图：*face_circle_inner_detected.png*
- 中间圆内圈-失败图：*face_circle_inner_fail.png*
- 中间圆内圈-成功图：*face_circle_inner_ok.png*
- 中间圆内圈-处理中时旋转图：*face_circle_inner_processing.png*
- 中间圆外圈-底图：*face_circle_outer_bg.png*
- 中间圆外圈-旋转图：*face_circle_outer_detected.png*
- 提示动画圆圈底图：*face_guide_bg.png*

活体环节自定义文案颜色

操作步骤

- Android

  1. 在*theme.json*修改颜色。
  2. 将*theme.json*放在*assets/facetheme/*下。

- iOS

  1. 在*theme.json*修改颜色。
  2. 将*theme.json*放在*FaceTheme.bundle*的*images*目录下。
  3. 将*FaceTheme.bundle*加入项目中。

原始配置说明

原始*theme.json*可在*FaceTheme.bundle*中的*images*目录中找到， 说明如下：

- Key说明
  - 错误文字颜色（检验不通过...）：color_error_text
  - 屏幕下方按钮文字颜色：color_button_text
  - 动作提示语文字颜色（请张下嘴...）：color_prompt_text
  - 导航首页提示文字颜色（请本人操作...）：color_nav_text
  - 悬浮提示文字颜色（请把脸部移入框中...）：color_tip_text
- 值说明：值为10进制整型，由16进制颜色（0xAARRGGBB）转化成10进制的整型，如0xFFFF5000对应4294922240。
- *theme.json*示例

```
{
"color_error_text":4294922240,
"color_button_text":4294967295,
"color_prompt_text":4281545523,
"color_nav_text":4281545523,
"color_tip_text":4282204485
}
```

2.2.5 SDK Release Notes

2.3 服务端接入

2.3.1 服务端接入准备

本文介绍了在服务端接入活体人脸验证时要完成的准备工作。

背景信息

单击查看老版本接入文档说明。

操作步骤

1. 前往阿里云官网注册账号。如果已有注册账号，请跳过此步骤。
2. 对该账号进行#unique_5。如果已经是企业账号，请跳过此步骤。

3. 打开云盾实人认证产品页面，单击立即开通，开通实人认证服务。
4. 在接入之前，可以根据业务上的需求，先了解#unique_6/unique_6_Connect_42_section_13r_whe_p8p，选择可以满足业务需求的认证方案。
5. 登录阿里云实人认证控制台，在接入及设置页面，创建场景并进行相应的流程配置，具体操作见#unique_7/unique_7_Connect_42_section_gbr_hh7_02f。
6. 根据#unique_8，了解客户端和服务端的交互流程，及各自需要做的事情。
7. 获取AccessKey。

   阿里云api的调用需要使用AccessKey来校验请求数据的安全性，接入方需要在AccessKey管理页面获取AccessKey。但因为阿里云账号的 AccessKey 具有所有云产品API的访问权限，一旦泄露将导致极大的安全风险，所以强烈建议您根据最小权限原则创建并使用RAM子用户来进行API访问和控制台操作，具体参见RAM子账号接入。
8. API调用方式。获取到AccessKey之后，参考API调用方式了解请求的结构和公共参数。
9. 服务端API。

API	描述
#unique_10（DescribeVerifyToken）	获取认证token。适用于SDK+服务端接入、H5+服务端接入的认证方案，在发起认证时调用。
#unique_11（DescribeVerifyResult）	查询认证结果。适用于SDK+服务端接入、H5+服务端接入的认证方案，认证后查询认证结果。

10. 接入具体的服务端API接口时，推荐使用我们的服务端SDK开发部。服务端SDK开发包中有对应的示例代码，便于接入方接入和调试。
11. 阿里云也提供了一个OpenAPI调用工具，可供接入方调试实人认证服务端API。

2.3.2 RAM子账号接入

实人认证服务端API支持您通过RAM子账号的方式进行调用。要使用RAM子账号调用实人认证API，您需要创建子账号并完成授权。

操作步骤

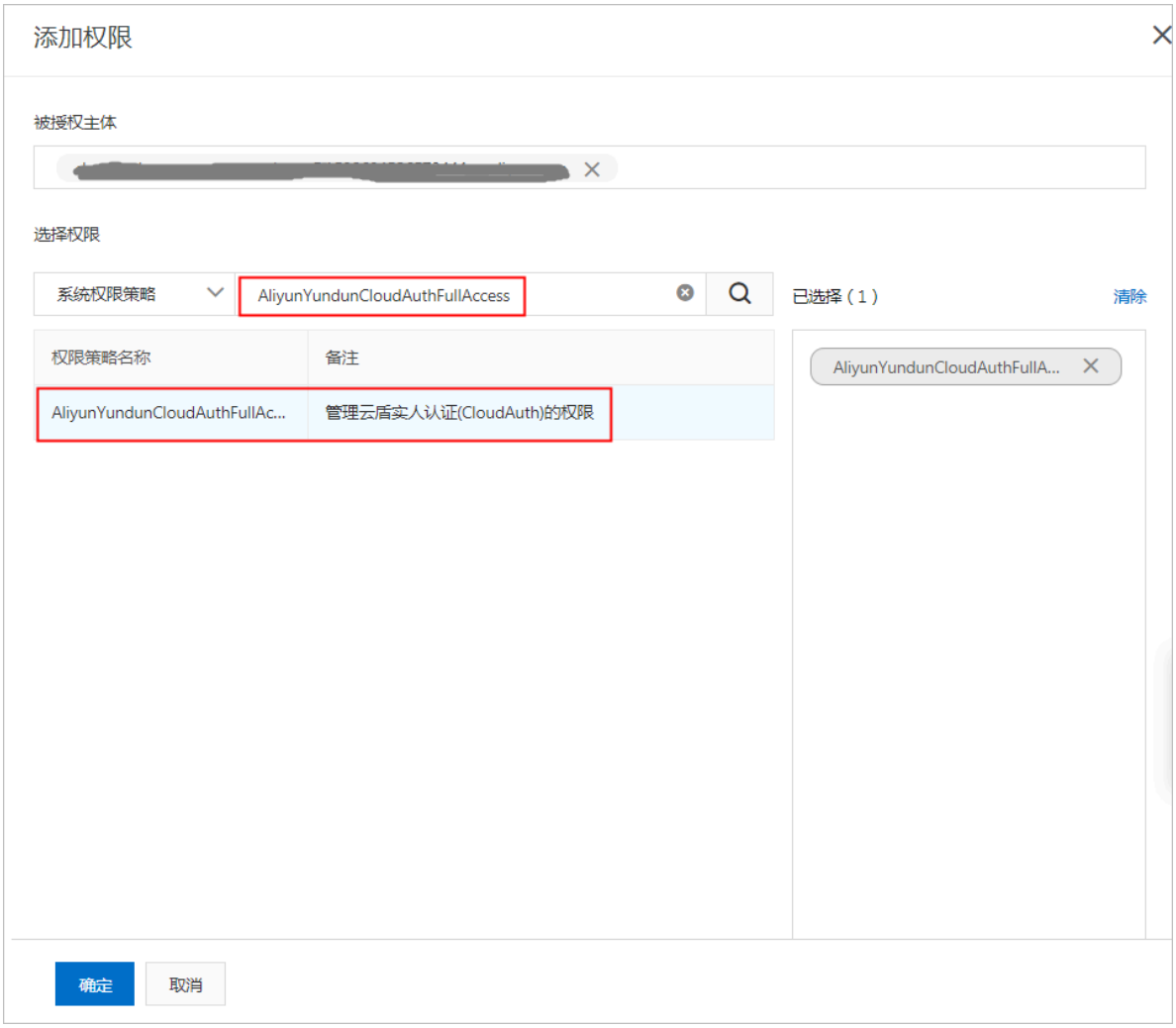
1. 登录RAM控制台，创建子账号， 并选择生成AccessKeyID和AccessKeySecret。创建后请妥善保存该AccessKeyID和AccessKeySecret，后续使用SDK时需要提供。

   说明:

更多关于创建RAM子账号的操作，请参见#unique_25。

2. 完成子账号授权。只有完成子账号授权，您的子账号才能够调用相关API。您需要向子账号授权以下系统策略权限：AliyunYundunCloudAuthFullAccess。

> **说明:**
> 更多关于子账号授权的操作，请参见#unique_26。



3. 给 RAM 子用户创建 AccessKey，具体请参见#unique_27。

2.3.3 API调用方式

对实人认证服务端API接口调用是通过向API的服务端地址发送HTTPS GET/POST请求，并按照接口说明在请求中加入相应请求参数来完成的；根据请求的处理情况，系统会返回处理结果。

请求结构

服务地址： 实人认证服务端API的服务接入地址为：`cloudauth.aliyuncs.com`。

通信协议： 为了保证安全性，实人认证只支持通过HTTPS通道进行请求通信。

请求方法： 支持HTTPS GET/POST方法发送请求，这种方式下请求参数需要包含在请求的URL中。

请求参数： 每个请求的参数都由两部分组成：

- 公共请求参数：用于指定API版本号、返回值类型、签名等，具体字段请参见下文。
- Action 及其特有的请求参数：Action参数对应具体要调用的接口名称（例如GetVerifyToken），其特有的请求参数即为该接口所要求的参数，这一部分的具体字段请参见各个API的说明文档。

字符编码： 请求及返回结果都使用 UTF-8 字符集进行编码。

说明:

为了便于进行开发，阿里云还提供Java、PHP、Python、.NET、Node.js、Go语言的SDK开发包，可以免去拼接 HTTPS 请求和对签名算法进行编码的麻烦。

公共参数

公共请求参数

公共请求参数是指每个接口都需要使用到的请求参数。

名称	类型	是否必需	描述
Format	String	否	返回值的类型，支持 JSON 与 XML，默认为 XML。
Version	String	是	API 版本号，为日期形式：YYYY-MM-DD，具体请参见API Release Notes。
AccessKeyId	String	是	阿里云颁发给用户的访问服务所用的密钥 ID。
Signature	String	是	用 AccessKeySecret 签名的结果串，关于签名的计算方法请参见RPC风格API的签名机制。
SignatureMethod	String	是	签名方式，目前支持 HMAC-SHA1。
Timestamp	String	是	请求的时间戳。日期格式按照 ISO8601 标准表示，并需要使用 UTC 时间 0 时区的值。格式为 YYYY-MM-DDThh:mm:ssZ。例如，2017-11-11T12:00:00Z（为北京时间 2017年11月11日20点0分0秒）。
SignatureVersion	String	是	签名算法版本，目前版本是1.0。
SignatureNonce	String	是	唯一随机数，用于防止网络重放攻击。用户在不同请求间要使用不同的随机数值。

请求示例

```
https://cloudauth.aliyuncs.com/?Format=xml
&Version=2017-11-17
&Signature=Pc5WB8gokVn0xfeu%2FZV%2BiNM1dgI%3D
&SignatureMethod=HMAC-SHA1
&SignatureNonce=15215528852396
&SignatureVersion=1.0
&AccessKeyId=key-test
&Timestamp=2017-11-11T12:00:00Z
&<[Action及其特有的请求参数]>
```

公共返回参数

用户发送的每次接口调用请求，无论成功与否，系统都会返回一个唯一识别码RequestId给用户。

- **XML返回示例**

```
<?xml version="1.0" encoding="UTF-8"?>
<!–结果的根结点-->
<接口名称+Response>
    <!–返回请求标签-->
    <RequestId>4C467B38-3910-447D-87BC-AC049166F216</RequestId>
    <!–返回结果数据-->
</接口名称+Response>
```

- **JSON返回示例**

```
{
    "RequestId": "4C467B38-3910-447D-87BC-AC049166F216",
    /* 返回结果数据 */
}
```

返回结果

调用 API 服务后返回数据采用统一格式，返回的 HTTP 状态码为 2xx，代表调用成功。返回 4xx 或 5xx 的 HTTP 状态码代表调用失败。

调用成功返回的数据格式主要有 XML 和 JSON 两种，外部系统可以在请求时传入参数来制定返回的数据格式，默认为 XML 格式。

本文档中的返回示例为了便于用户查看，做了格式化处理，实际返回结果是没有进行换行、缩进等处理的。

- **成功结果**
  - **XML返回示例（XML返回结果包括请求是否成功信息和具体的业务数据）**

```
<?xml version="1.0" encoding="UTF-8"?>
<!–结果的根结点-->
<接口名称+Response>
    <!–返回请求标签-->
```

```
    <RequestId>4C467B38-3910-447D-87BC-AC049166F216</RequestId>
    <!—返回结果数据-->
</接口名称+Response>
```

- JSON返回示例

```
{
    "RequestId": "4C467B38-3910-447D-87BC-AC049166F216",
    /* 返回结果数据 */
}
```

· 错误结果

调用接口出错后，将不会返回结果数据。调用方可根据接口文档中的错误码说明来定位错误原因。

当调用出错时，HTTP 请求返回一个 4xx 或 5xx 的 HTTP 状态码。返回的消息体中是具体的错误代码及错误信息。另外还包含一个全局唯一的请求 ID RequestId和一个您该次请求访问的站点 ID HostId。在调用方找不到错误原因时，可以联系阿里云客服，并提供该HostId和RequestId，以便我们尽快帮您解决问题。

- XML返回示例

```
<?xml version="1.0" encoding="UTF-8"?>
<Error>
<RequestId>8906582E-6722-409A-A6C4-0E7863B733A5</RequestId>
   <HostId>cloudauth.aliyuncs.com</HostId>
   <Code>UnsupportedOperation</Code>
   <Message>The specified action is not supported.</Message>
</Error>
```

- JSON返回示例

```
{
    "RequestId": "8906582E-6722-409A-A6C4-0E7863B733A5",
    "HostId": "cloudauth.aliyuncs.com",
    "Code": "UnsupportedOperation",
    "Message": "The specified action is not supported."
}
```

2.3.4 发起认证请求

调用本接口发起认证请求。

单击查看老版本接入文档说明。

接入准备

接入该API时，请确保已经完成相应的准备工作，具体参见#unique_15。

描述

接口名：DescribeVerifyToken

请求方法：HTTPS POST 和 GET。

接口说明：每次开始认证前通过本接口获取认证Token（VerifyToken），用来串联认证请求中的各个接口。

适用范围：该接口适用于SDK+服务端接入、H5+服务端接入的认证方案。

图片地址：需使用HTTP/HTTPS地址，公网可访问的HTTP/HTTPS地址。例如，`http://image-demo.img-cn-hangzhou.aliyuncs.com/example.jpg`。

> **说明:**
>
> - 不支持本地图片的相对路径或绝对路径。
> - 单张图片大小请控制在2M内，避免算法拉取超时。
> - 图片中人脸区域的大小至少64*64像素。
> - 单个请求的Body有8M的大小限制，请计算好请求中所有图片和其他信息的大小，不要超限。

请求参数

名称	类型	参数位置	是否必需	描述
Action	String	query	是	要执行的操作。取值：DescribeVerifyToken。
RegionId	String	query	是	服务所在地域。取值：cn-hangzhou。
BizType	String	query	是	使用实人认证服务的业务场景标识。请先参见#unique_7在控制台完成创建。
BizId	String	query	是	标识一次认证任务的唯一ID，不超过64字符。针对一次认证任务，系统支持无限次发起提交，直到最终认证通过，该任务完结。 **说明:** 发起不同的认证任务时需要更换不同的BizId。
UserId	String	query	否	C用户的ID，如C用户账号ID。
Name	String	query	否	姓名。

名称	类型	参数位置	是否必需	描述
IdCardNumber	String	query	否	身份证号。
IdCardFrontImageUrl	String	query	否	身份证人像面图片的HTTP/HTTPS链接。
IdCardBackImageUrl	String	query	否	身份证国徽面图片的HTTP/HTTPS链接。
FaceRetainedImageUrl	String	query	否	人像留底照片的HTTP/HTTPS链接。
IdImageUrl	String	query	否	二代身份证芯片中存储的头像照片的HTTP/HTTPS链接。
PassedRedirectUrl	String	query	否	认证通过重定向URL，H5方案适用。
FailedRedirectUrl	String	query	否	认证失败重定向URL，H5方案适用。
CallbackUrl	String	query	否	服务端回调URL。
CallbackSeed	String	query	否	用于回调签名，使用callbackUrl时callbackSeed必填。

根据选择的认证方案不同，所需传入的字段不太一样，具体可以参见下述表格中的说明：

认证方案	Name字段	IdCardNumber字段	IdCardFrontImageUrl字段	IdCardBackImageUrl字段	FaceRetainedImageUrl字段	IdImageUrl字段
FVBioOnly	否	否	否	否	是	否
FDBioOnly	否	否	否	否	否	否

返回参数

名称	类型	是否必须	描述
VerifyToken	字符串	是	认证token。
VerifyPageUrl	字符串	否	H5认证页面的入口链接，适用于H5方案，其他认证方案不需要关注。

示例

关于使用服务端SDK开发包的示例，请参见各语言服务端SDK文档中的认证方案示例：

- Java SDK

- PHP SDK
- Python SDK
- .NET SDK
- Node.js SDK
- Go SDK

请求示例

```
https://cloudauth.aliyuncs.com/?Action= DescribeVerifyToken
&RegionId=cn-hangzhou
&BizType=RPBasicTest
&BizId=39ecf51e-2f81-4dc5-90ee-ff86125be683
&<[公共请求参数]>
```

返回示例

- XML格式

```
<?xml version='1.0'
encoding='UTF-8'?>
<DescribeVerifyTokenResponse>
    <Data>
       <VerifyToken>c302c0797679457685410ee51a5ba375</VerifyToken>
    </Data>
</VerifyMaterialResponse>
```

- JSON格式

```
{
"VerifyToken": "c302c0797679457685410ee51a5ba375",
}
```

2.3.5 查询认证结果

调用本接口查询认证结果。

单击查看老版本接入文档说明。

接入准备

接入该API时，请确保已经完成相应的准备工作，具体参见#unique_15。

描述

接口名：DescribeVerifyResult

请求方法：HTTPS POST 和 GET。

接口说明：当接入方移动端收到回调之后，其服务端可以调用此接口，来获取相应的认证状态和认证资料。

适用范围：该接口适用于SDK+服务端接入、H5+服务端接入的认证方案。

请求参数

名称	类型	是否必需	描述
BizId	String	是	对应于DescribeVerifyToken接口里传入的BizId。

返回参数

名称	类型	是否必需	描述
VerifyStatus	Integer	是	认证状态，取值： · -1：未认证 · 1：认证通过 · 2~n：各种不同原因导致的认证不通过 详细描述参见认证状态说明。
Material	Map	否	认证材料。具体结构描述参见Material。
AuthorityComparisonScore	Float	是	认证过程中所提交的人脸照片和权威数据的比对分，取值范围为[0,100]。置信度阈值请参见： · 误识率0.001%时，对应阈值95。 · 误识率0.01%时，对应阈值90。 · 误识率为0.1%时，对应阈值80。 · 误识率为1%时，对应阈值为60。 说明： 该字段只表示人脸与权威数据的比对结果，是个参考分，通常不建议业务上仅以该分数作为是否通过的标准。认证的综合结果请参见VerifyStatus字段。VerifyStatus的结果综合了人脸与权威数据的比对和其他多种策略，可以提高安全水位。

表 2-1: Material

名称	类型	是否必需	描述
FaceImageUrl	String	是	人像正面照图片HTTP/HTTPS链接。链接地址5分钟内有效，建议业务上进行转存以免影响使用。
IdCardName	String	否	姓名。
IdCardNumber	String	否	身份证号。

名称	类型	是否必需	描述
IdCardInfo	Map	否	身份证信息的OCR结果。具体结构描述参见表 2-2: IdCardInfo。 说明: 认证过程中如果没有身份证正面和身份证反面，则不会返回身份证信息的OCR结果。认证过程中如果有身份证正面和身份证反面，也不保证一定会返回身份证上所有的信息，因身份证拍摄问题等引起的OCR无法识别时，OCR信息就会不全，建议接入方业务上不强依赖身份证OCR信息。

表 2-2: IdCardInfo

名称	类型	是否必需	描述
FrontImageUrl	String	是	身份证人像面图片HTTP/HTTPS链接。链接地址5分钟内有效，建议业务上进行转存以免影响使用。
BackImageUrl	String	是	身份证国徽面图片HTTP/HTTPS链接。链接地址5分钟内有效，建议业务上进行转存以免影响使用。
Name	String	否	姓名。
Number	String	否	身份证号。
Sex	String	否	性别。取值: · 男 · 女
Nationality	String	否	民族。
Address	String	否	地址。
StartDate	String	否	证件有效期开始时间。格式为：yyyy-MM-dd。
EndDate	String	否	证件有效期结束时间。格式为：yyyy-MM-dd。
Authority	String	否	签发机构。

表 2-3: 认证状态说明

VerifyStatus	原因说明
-1	未认证。这种状态一般是端上没有成功提交比如用户中途退出认证流程。
1	认证通过。

VerifyStatus	原因说明
9	认证不通过，可能原因：非账户本人操作等可能。
10	认证不通过，可能原因：非同人操作等可能。

示例

关于使用服务端SDK开发包的示例，请参见以下具体SDK 开发包，使用文档中的认证方案说明：Java、PHP、Python、.NET、Node.js、Go。

请求示例

```
https://cloudauth.aliyuncs.com/?Action=VerifyMaterials
&RegionId=cn-hangzhou
&BizType=FVBioOnlyTest
&BizId=39ecf51e-2f81-4dc5-90ee-ff86125be683
&<[公共请求参数]>
```

返回示例

- XML格式

```
<?xml version='1.0' encoding='UTF-8'?>
<VerifyMaterialsResponse>
    <Data>
<VerifyStatus>1</VerifyStatus>
       <Material>
           <FaceImageUrl>http://image-demo.img-cn-hangzhou.aliyuncs.
com/example.jpg
           </FaceImageUrl>
       </Material>
    </Data>
</VerifyMaterialsResponse>
```

- JSON格式

```
{
      "VerifyStatus": "1",
      "Material": {
        "FaceImageUrl": "http://image-demo.img-cn-hangzhou.aliyuncs.
com/example.jpg",
        }
      },
```

```
}
```

2.3.6 API Release Notes

查看历次API更新说明。

API版本号	发布说明
2019-03-07	更新内容如下： · 发布新接口：发起认证请求（DescribeVerifyToken）。 · 发布新接口：获取认证结果（DescribeVerifyResult）。 · 发布新接口：提交认证（VerifyMaterial）。
2018-09-16	更新内容如下： · 提交认证资料接口（SubmitMaterials）增加一个返回参数（AuditConclusions）。 · 提交认证资料接口（SubmitMaterials）增加一个返回参数（AuthorityComparisonScore）。 · 查询认证状态接口（GetStatus）增加一个返回参数（AuthorityComparisonScore）。 · 发布提交认证接口（SubmitVerification）。
2018-08-07	更新内容如下： · 版本 1.1.3 的问题修复。 · 发起认证请求接口（GetVerifyToken）增加一个可选入参（VerifyConfigs）。
2018-07-03	更新内容如下： · 发起认证请求接口（GetVerifyToken）增加认证页面 URL（CloudauthPageUrl）字段。 · 增加人脸属性检测接口（DetectFaceAttributes）。
2018-05-04	更新内容如下： 获取认证资料接口（GetMaterials）增加身份证 OCR 识别的证件有效期起始日期（IdCardStartDate）、民族（EthnicGroup）字段。
2017-11-17	更新内容如下： 查询认证状态接口（GetStatus）增加认证不通过原因（AuditConclusions）字段。

API版本号	发布说明
2017-10-10	更新内容如下： · 人脸比对验证接口CompareFaces首次发布。 · 查询认证状态接口GetStatus首次发布。 · 发起认证请求接口GetVerifyToken首次发布。 · 提交认证资料接口SubmitMaterials首次发布。 · 获取认证资料接口GetMaterials首次发布。

2.4 服务端SDK开发包

2.4.1 Java SDK

本文介绍了如何使用阿里云实人认证服务的Java SDK，具体包括SDK的获取和安装方法以及SDK代码示例。

单击查看老版本接入文档说明。

获取地址

您需要引入两个SDK，包括*aliyun-java-sdk-core*（阿里云核心SDK）和*aliyun-java-sdk-cloudauth*（实人认证SDK）。每个SDK都提供了Maven Repository、Central Repository、GitHub的获取方式，您可以选择合适的方式获取SDK。

· *aliyun-java-sdk-core*：从以下方式中单击选择一种方式。

- mvnrepository
- maven.org
- GitHub

> 说明：
> 若您使用的是*aliyun-java-sdk-core* 4.0.0~4.0.2 版本，那么在调用HTTPS接口时需要在`profile`中额外添加以下内容：`profile.getHttpClientConfig().setIgnoreSSLCerts(true);`。

· *aliyun-java-sdk-cloudauth*：从以下方式中单击选择一种方式。

- mvnrepository
- maven.org
- GitHub

安装说明

方法1：使用Maven（推荐）

如果您使用Maven管理Java项目，可以通过在*pom.xml*文件中添加Maven依赖：

```
<dependency>
    <groupId>com.aliyun</groupId>
    <artifactId>aliyun-java-sdk-core</artifactId>
    <version>4.4.3</version>
</dependency>
<dependency>
    <groupId>com.aliyun</groupId>
    <artifactId>aliyun-java-sdk-cloudauth</artifactId>
    <version>2.0.0</version>
</dependency>
```

说明:

`version`**的值以SDK获取地址中的最新版本为准。**

方法2：在集成开发环境（IDE）中导入jar文件

- **Eclipse安装**
  1. **将下载的*aliyun-java-sdk-xxx.jar*文件复制到您的项目文件夹中。**
  2. **在Eclipse中打开您的项目，右键单击该项目，单击Properties。**
  3. **在弹出的对话框中，单击Java Build Path > Libraries > Add JARs添加下载的JAR文件。**
  4. **单击Apply and Close。**
- **IntelliJ 安装**
  1. **将下载的*aliyun-java-sdk-xxx.jar*文件复制到您的项目文件夹中。**
  2. **在IntelliJ中打开您的项目，在菜单栏中单击File > Project > Structure。**
  3. **单击Apply，然后单击OK。**

RPBasic、RPManual、FDBioOnly认证方案示例代码

```
DefaultProfile profile = DefaultProfile.getProfile(
                    "cn-hangzhou",           // 您的可用区ID
                    "AccessKey",       // 您的Access Key ID
                    "AccessSecret"); // 您的Access Key Secret
DefaultProfile.addEndpoint("cn-hangzhou", "cn-hangzhou", "Cloudauth",
 "cloudauth.aliyuncs.com");
IAcsClient client = new DefaultAcsClient(profile);

DescribeVerifyTokenRequest request = new DescribeVerifyTokenRequest();
request.setBizId("认证ID，由接入方指定，发起不同的认证任务需要更换不同的认证ID
");
request.setBizType("实人认证控制台上创建场景时对应的场景标识"); //创建方法请参
见https://help.aliyun.com/document_detail/127885.htm

DescribeVerifyTokenResponse response = client.getAcsResponse(request);
```

```
String verifyToken = response.getVerifyToken();

//2. 接入方服务端将token传递给接入方无线客户端
//3. 接入方无线客户端用token调起无线认证SDK
//4. 用户按照由无线认证SDK组织的认证流程页面的指引，提交认证资料
//5. 认证流程结束退出无线认证SDK，进入客户端回调函数
//6. 接入方服务端获取认证状态和认证资料(注：客户端无线认证SDK回调中也会携带认证状
态，但建议以服务端调接口获取的为准进行业务上的判断和处理)
DescribeVerifyResultRequest verifyResultRequest = new DescribeVe
rifyResultRequest();
verifyResultRequest.setBizId("调用GetVerifyToken时传入的bizId");
verifyResultRequest.setBizType("调用GetVerifyToken时传入的bizType");
DescribeVerifyResultResponse verifyResultResponse = client.getAcsResp
onse(verifyResultRequest);
```

RPBioID、RPBioOnlyPro、RPBioOnly、RPH5BioOnly认证方案示例代码

```
DefaultProfile profile = DefaultProfile.getProfile(
                    "cn-hangzhou",           // 您的可用区ID
                    "AccessKey",       // 您的Access Key ID
                    "AccessSecret"); // 您的Access Key Secret
DefaultProfile.addEndpoint("cn-hangzhou", "cn-hangzhou", "Cloudauth",
 "cloudauth.aliyuncs.com");
IAcsClient client = new DefaultAcsClient(profile);

DescribeVerifyTokenRequest request = new DescribeVerifyTokenRequest();
request.setBizId("认证ID，由接入方指定，发起不同的认证任务需要更换不同的认证ID
 ");
request.setBizType("实人认证控制台上创建场景时对应的场景标识"); //创建方法请参
见https://help.aliyun.com/document_detail/127885.htm
request.setName("用户正确的姓名");
request.setIdCardNumber("用户正确的身份证号");

DescribeVerifyTokenResponse response = client.getAcsResponse(request);
String verifyToken = response.getVerifyToken();
// verifyPageUrl仅在RPH5BioOnly认证方案下返回
String verifyPageUrl = response.getVerifyPageUrl();

//2. 接入方服务端将token传递给接入方无线客户端
//3. 接入方无线客户端用token调起无线认证SDK
//4. 用户按照由无线认证SDK组织的认证流程页面的指引，提交认证资料
//5. 认证流程结束退出无线认证SDK，进入客户端回调函数
//6. 接入方服务端获取认证状态和认证资料(注：客户端无线认证SDK回调中也会携带认证状
态，但建议以服务端调接口获取的为准进行业务上的判断和处理)
DescribeVerifyResultRequest verifyResultRequest = new DescribeVe
rifyResultRequest();
verifyResultRequest.setBizId("调用GetVerifyToken时传入的bizId ");
verifyResultRequest.setBizType("调用GetVerifyToken时传入的bizType ");
DescribeVerifyResultResponse verifyResultResponse = client.getAcsResp
onse(verifyResultRequest);
```

FVBioOnly认证方案示例代码

```
DefaultProfile profile = DefaultProfile.getProfile(
                    "cn-hangzhou",           // 您的可用区ID
                    "AccessKey",       // 您的Access Key ID
                    "AccessSecret"); // 您的Access Key Secret
DefaultProfile.addEndpoint("cn-hangzhou", "cn-hangzhou", "Cloudauth",
 "cloudauth.aliyuncs.com");
IAcsClient client = new DefaultAcsClient(profile);

DescribeVerifyTokenRequest request = new DescribeVerifyTokenRequest();
```

```
request.setBizId("认证ID，由接入方指定，发起不同的认证任务需要更换不同的认证ID
 ");
request.setBizType("实人认证控制台上创建场景时对应的场景标识"); //创建方法请参
见https://help.aliyun.com/document_detail/127885.htm

request.setFaceRetainedImageUrl("公网可访问的图片http/https链接");

DescribeVerifyTokenResponse response = client.getAcsResponse(request);
String verifyToken = response.getVerifyToken();

//2. 接入方服务端将token传递给接入方无线客户端
//3. 接入方无线客户端用token调起无线认证SDK
//4. 用户按照由无线认证SDK组织的认证流程页面的指引，提交认证资料
//5. 认证流程结束退出无线认证SDK，进入客户端回调函数
//6. 接入方服务端获取认证状态和认证资料(注：客户端无线认证SDK回调中也会携带认证状
态，但建议以服务端调接口获取的为准进行业务上的判断和处理)
DescribeVerifyResultRequest verifyResultRequest = new DescribeVe
rifyResultRequest();
verifyResultRequest.setBizId("调用GetVerifyToken时传入的bizId ");
verifyResultRequest.setBizType("调用GetVerifyToken时的bizType");
DescribeVerifyResultResponse verifyResultResponse = client.getAcsResp
onse(verifyResultRequest);
```

RPMin认证方案示例代码

RPMin认证方案的人脸照片入参，支持公网可访问的HTTP/HTTPS链接，也支持接入方使用实人认证提供的SDK将base64的图片上传到实人认证OSS Bucket后生成https链接。

说明:

如果您的业务需要使用上传SDK，请提交工单联系我们获取。

```
DefaultProfile profile = DefaultProfile.getProfile(
                    "cn-hangzhou",           // 您的可用区ID
                    "AccessKey",       // 您的Access Key ID
                    "AccessSecret"); // 您的Access Key Secret
DefaultProfile.addEndpoint("cn-hangzhou", "cn-hangzhou", "Cloudauth",
 "cloudauth.aliyuncs.com");
IAcsClient client = new DefaultAcsClient(profile);

//若接入方的人脸图片是本地资源，则可以使用实人认证提供的上传SDK将图片直传到实人认证
OSS Bucket并获取到图片地址。如果您的业务需要使用上传SDK，请提交工单联系我们获取。
CloudAuthClientUploader uploader = CloudAuthClientUploader.getClentUp
loader(client); // 获取上传oss的实例
String faceImageUrl = uploader.uploadBase64("待上传的base64图片资
源"); // 上传oss并获取图片链接

VerifyMaterialRequest verifyMaterialRequest = new VerifyMaterialReques
t();
verifyMaterialRequest.setBizId("认证ID，由接入方指定，发起不同的认证任务需要
更换不同的认证ID ");
verifyMaterialRequest.setBizType("实人认证控制台上创建场景时对应的场景标
识"); //创建方法请参见https://help.aliyun.com/document_detail/127885.htm

verifyMaterialRequest.setFaceImageUrl(faceImageUrl); // faceImageUrl
可以是通过直传OSS获取到的链接，也可以是接入方公网可访问的人脸图片链接，支持http/
https
verifyMaterialRequest.setName("用户正确的姓名");
verifyMaterialRequest.setIdCardNumber("用户正确的身份证号");
```

```
VerifyMaterialResponse verifyMaterialResponse = client.getAcsResponse(
verifyMaterialRequest);
int statusCode = verifyMaterialResponse.getVerifyStatus(); //同步返回认
证状态和相应材料
```

人脸比对验证示例

```
//创建DefaultAcsClient实例并初始化
DefaultProfile profile = DefaultProfile.getProfile(
        "cn-hangzhou",              //默认
        "YourAccessKeyID",          //您的Access Key ID
        "YourAccessKeySecret");     //您的Access Key Secret
IAcsClient client = new DefaultAcsClient(profile);
//创建API请求并设置参数
//CompareFaces接口文档: https://help.aliyun.com/document_detail/59317.
html
CompareFacesRequest request = new CompareFacesRequest();
request.setMethod(MethodType.POST);
//传入图片资料，请控制单张图片大小在 2M 内，避免拉取超时
request.setSourceImageType("FacePic");
request.setSourceImageValue("base64://iVBORw0KGgoA..."); //base64方式上
传图片，格式为"base64://图片base64字符串"，以"base64://"开头且图片base64字符
串去掉头部描述(如"")，并注意控制接口请求的Body在8M以内
request.setTargetImageType("FacePic"); //若为身份证芯片照则传"IDPic"
request.setTargetImageValue("http://image-demo.img-cn-hangzhou.
aliyuncs.com/example.jpg"); //http方式上传图片，此http地址须可公网访问
//发起请求并处理异常
try {
    CompareFacesResponse response = client.getAcsResponse(request);
    //后续业务处理
} catch (ServerException e) {
    e.printStackTrace();
} catch (ClientException e) {
    e.printStackTrace();
}
//常见问题: https://help.aliyun.com/document_detail/57640.html
```

离线人脸识别SDK下载示例

```
DefaultProfile profile = DefaultProfile.getProfile(
"cn-hangzhou", // 可用区域id，目前只支持cn-hangzhou
"your access key id", // 您的Access Key ID
"your access secret"); // 您的Access Key Secret
DefaultProfile.addEndpoint("cn-hangzhou", "Cloudauth", "cloudauth.
aliyuncs.com"); //手动添加域名
client = new DefaultAcsClient(profile);

try {
CreateVerifySDKRequest createRequest = new CreateVerifySDKRequest();
createRequest.setAppUrl("https://app"); //app的可访问地址
CreateVerifySDKResponse createResponse = client.getAcsResponse(
createRequest);
String taskId = createResponse.getTaskId(); //获取生成sdk任务的taskId
String sdkUrl = null;
do {
//使用taskId轮询结果，一般生成可以在1分钟内完成
Thread.sleep(TimeUnit.SECONDS.toMillis(15));
DescribeVerifySDKRequest request = new DescribeVerifySDKRequest();
request.setTaskId(taskId);
DescribeVerifySDKResponse describeVerifySDKResponse = null;
```

```
describeVerifySDKResponse = client.getAcsResponse(request);
sdkUrl = describeVerifySDKResponse.getSdkUrl();
} while (sdkUrl == null || sdkUrl.isEmpty());
//sdkUrl为生成的sdk可访问链接，下载后进行集成
} catch (ClientException e) {
//生成异常
} catch (InterruptedException e) {

}
```

离线人脸识别SDK获取授权key示例

```
DefaultProfile profile = DefaultProfile.getProfile(
"cn-hangzhou", // 可用区域id，目前只支持cn-hangzhou
"your access key id", // 您的Access Key ID
"your access secret"); // 您的Access Key Secret
DefaultProfile.addEndpoint("cn-hangzhou", "Cloudauth", "cloudauth.
aliyuncs.com"); //手动添加域名
client = new DefaultAcsClient(profile);

//发起获取授权key的请求
CreateAuthKeyRequest request = new CreateAuthKeyRequest();
request.setTest(Boolean.FALSE); //测试标识
request.setAuthYears(1);//授权年限
request.setBizType("biz type"); //业务类型
request.setUserDeviceId("device id"); //可自定义的用户设备id
CreateAuthKeyResponse createAuthKeyResponse = client.getAcsResponse(
request);
String authKey = createAuthKeyResponse.getAuthKey();
//获取到授权key调用离线人脸识别SDK的initWithToken进行设备激活
```

2.4.2 PHP SDK

本文介绍了如何使用阿里云实人认证服务的PHP SDK。

获取地址

单击前往GitHub。

使用说明

参见GitHub中的说明。

示例代码

```
<?php
require_once 'vendor/autoload.php';

use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;

/**
 * 创建一个全局客户端
 */
```

```
AlibabaCloud::accessKeyClient('xxx', 'xxxxxx')->regionId('cn-hangzhou
')->asDefaultClient();
```

发起认证请求DescribeVerifyToken示例代码

- RPBasic、RPManual、FDBioOnly认证方案

```
try {
    // 访问产品 APIs
    $request = AlibabaCloud::Cloudauth()->V20190307()-
>DescribeVerifyToken();
    $result = $request->withBizType('实人认证控制台上创建场景时对应的场景
标识') //创建方法参见#unique_36
                      ->withBizId('认证ID，由接入方指定，发起不同的认证任
务需要更换不同的认证ID')
                      ->connectTimeout(10)
                      ->timeout(10)
                      ->request();
    // Convert the result to an array and print
    print_r($result->toArray());
} catch (ClientException $e) {
    // Print the error message
    echo $e->getErrorMessage() . PHP_EOL;
} catch (ServerException $e) {
    // Print the error message
    echo $e->getErrorMessage() . PHP_EOL;
    print_r($e->getResult()->toArray());
}
```

- RPBioID、RPBioOnlyPro、RPBioOnly、RPH5BioOnly认证方案

```
try {
    // 访问产品 APIs
    $request = AlibabaCloud::Cloudauth()->V20190307()-
>DescribeVerifyToken();

    $result = $request->withBizType('实人认证控制台上创建场景时对应的场景
标识') //创建方法参见#unique_36
                      ->withBizId('认证ID，由接入方指定，发起不同的认证任
务需要更换不同的认证ID')
                      ->withIdCardNumber('用户正确的身份证号')
                      ->withName('用户正确的姓名')
                      ->connectTimeout(10)
                      ->timeout(10)
                      ->request();
    // Convert the result to an array and print
    print_r($result->toArray());
} catch (ClientException $e) {
    // Print the error message
    echo $e->getErrorMessage() . PHP_EOL;
} catch (ServerException $e) {
    // Print the error message
    echo $e->getErrorMessage() . PHP_EOL;
    print_r($e->getResult()->toArray());
}
```

- FVBioOnly认证方案

```
try {
    // 访问产品 APIs
```

```
    $request = AlibabaCloud::Cloudauth()->V20190307()-
>DescribeVerifyToken();

    $result = $request->withBizType('实人认证控制台上创建场景时对应的场景
标识') //创建方法参见#unique_36
                      ->withBizId('认证ID，由接入方指定，发起不同的认证任
务需要更换不同的认证ID')
                      ->withFaceRetainedImageUrl('公网可访问的图片http/
https链接')
                      ->connectTimeout(10)
                      ->timeout(10)
                      ->request();
    // Convert the result to an array and print
    print_r($result->toArray());
} catch (ClientException $e) {
    // Print the error message
    echo $e->getErrorMessage() . PHP_EOL;
} catch (ServerException $e) {
    // Print the error message
    echo $e->getErrorMessage() . PHP_EOL;
    print_r($e->getResult()->toArray());
}
```

查询认证结果DescribeVerifyResult示例代码

```
try {
    // 访问产品 APIs
    $request = AlibabaCloud::Cloudauth()->V20190307()-
>DescribeVerifyResult();

    $result = $request->withBizType('实人认证控制台上创建场景时对应的场景标
识') //创建方法参见#unique_36
                      ->withBizId('认证ID，由接入方指定，发起不同的认证任务
需要更换不同的认证ID')
                      ->connectTimeout(10)
                      ->timeout(10)
                      ->request();
    // Convert the result to an array and print
    print_r($result->toArray());
} catch (ClientException $e) {
    // Print the error message
    echo $e->getErrorMessage() . PHP_EOL;
} catch (ServerException $e) {
    // Print the error message
    echo $e->getErrorMessage() . PHP_EOL;
    print_r($e->getResult()->toArray());
}
```

RPMin认证方案示例代码

```
try {
    // 访问产品 APIs
    $request = AlibabaCloud::Cloudauth()->V20190307()-
>VerifyMaterial();

    $result = $request->withBizType('实人认证控制台上创建场景时对应的场景标
识') //创建方法参见#unique_36
                      ->withBizId('认证ID，由接入方指定，发起不同的认证任务
需要更换不同的认证ID')
                      ->withName('用户正确的姓名')
                      ->withIdCardNumber('用户正确的身份证号')
```

```
                        ->withFaceImageUrl('公网可访问的图片http/https链
接')
                        ->connectTimeout(10)
                        ->timeout(10)
                        ->request();
    // Convert the result to an array and print
    print_r($result->toArray());
} catch (ClientException $e) {
    // Print the error message
    echo $e->getErrorMessage() . PHP_EOL;
} catch (ServerException $e) {
    // Print the error message
    echo $e->getErrorMessage() . PHP_EOL;
    print_r($e->getResult()->toArray());
}
```

人脸比对示例代码

```
try {
    // 访问产品 APIs
    $request = AlibabaCloud::Cloudauth()->V20190307()->CompareFaces();

    $result = $request->withSourceImageType('FacePic')
                        ->withTargetImageType('FacePic')
                        ->withSourceImageValue('公网可访问的图片http/https
链接')
                        ->withTargetImageValue('公网可访问的图片http/https
链接')
                        ->connectTimeout(10)
                        ->timeout(10)
                        ->request();
    // Convert the result to an array and print
    print_r($result->toArray());
} catch (ClientException $e) {
    // Print the error message
    echo $e->getErrorMessage() . PHP_EOL;
} catch (ServerException $e) {
    // Print the error message
    echo $e->getErrorMessage() . PHP_EOL;
    print_r($e->getResult()->toArray());
}
```

人脸属性检测示例代码

```
try {
    // 访问产品 APIs
    $request = AlibabaCloud::Cloudauth()->V20190307()->DetectFace
Attributes();

    $result = $request->withRetAttributes('facetype,headpose,age
') //具体的属性由接入方按需传入
                        ->withMaterialValue('公网可访问的图片http/https链
接')
                        ->withMaxFaceNum(1)
                        ->withDontSaveDB('false')
                        ->withClientTag('null')
                        ->withMaxNumPhotosPerCategory(1)
                        ->connectTimeout(10)
                        ->timeout(10)
                        ->request();
    // Convert the result to an array and print
    print_r($result->toArray());
```

```
} catch (ClientException $e) {
    // Print the error message
    echo $e->getErrorMessage() . PHP_EOL;
} catch (ServerException $e) {
    // Print the error message
    echo $e->getErrorMessage() . PHP_EOL;
    print_r($e->getResult()->toArray());
}
```

2.4.3 Python SDK

本文介绍了如何使用阿里云实人认证服务的Python SDK。

获取地址

您需要引入两个SDK，包括*aliyun-java-sdk-core*（阿里云核心SDK）和*aliyun-java-sdk-cloudauth*（实人认证SDK）。单击GitHub，通过GitHub获取需要的Python SDK。

使用说明

参见GitHub中的说明。

使用示例

完整示例正在准备中，请先参见Java SDK示例。

2.4.4 .NET SDK

本文介绍了如何使用阿里云实人认证服务的.NET SDK。

获取地址

您需要引入两个SDK，包括*aliyun-java-sdk-core*（阿里云核心SDK）和*aliyun-java-sdk-cloudauth*（实人认证SDK）。单击GitHub，通过GitHub获取需要的.NET SDK。

使用说明

参见GitHub中的说明。

使用示例

完整示例正在准备中，请先参见Java SDK示例。

2.4.5 Node.js SDK

本文介绍了如何使用阿里云实人认证服务的Node.js SDK。

获取地址

单击前往GitHub，基于核心SDK进行开发，使用RPC调用风格。

使用说明

参见GitHub中的说明。

使用示例

完整示例正在准备中，请先参见Java SDK示例。

2.4.6 Go SDK

本文介绍了如何使用阿里云实人认证服务的Go SDK。

获取地址

单击前往GitHub，使用泛化调用接口（CommonAPI）。

使用说明

参见GitHub中的说明。

使用示例

完整示例正在准备中，请先参见Java SDK示例。