

阿里云 实人认证 开发指南

文档版本：20181122

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。
<code>courier</code> 字体	命令。	执行 <code>cd /d C:/windows</code> 命令，进入Windows系统文件夹。
斜体	表示参数、变量。	<code>bae log list --instanceid Instance_ID</code>
[]或者[a b]	表示可选项，至多选择一个。	<code>ipconfig [-all -t]</code>
{ }或者{a b}	表示必选项，至多选择一个。	<code>swich {stand slave}</code>

目录

法律声明.....	I
通用约定.....	I
1 开发必读.....	1
2 服务端API.....	5
2.1 获取AccessKey.....	5
2.2 API调用方式.....	5
2.3 API Release Notes.....	8
2.4 API说明文档.....	9
2.4.1 查询认证状态.....	9
2.4.2 发起认证请求.....	12
2.4.3 提交认证资料.....	18
2.4.4 获取认证资料.....	23
2.4.5 人脸比对验证.....	28
2.4.6 人脸有效性检测.....	32
2.5 使用SDK开发包.....	37
2.5.1 Java SDK.....	37
2.5.2 PHP SDK.....	45
2.5.3 Python SDK.....	54
2.5.4 .NET SDK.....	54
2.5.5 Node.js SDK.....	55
2.5.6 Go SDK.....	55
3 无线认证SDK.....	56
3.1 下载无线SDK.....	56
3.2 Android集成.....	56
3.3 iOS集成.....	60
4 定向发布.....	64
4.1 提交认证.....	64
4.2 实人认证眼纹版开发文档.....	69

1 开发必读

实人认证/活体人脸验证



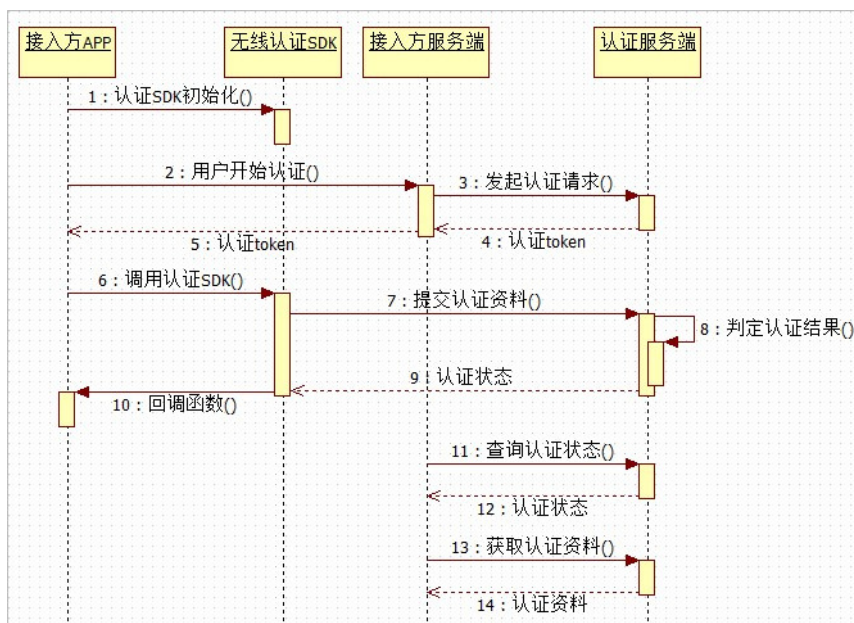
说明：

在正式进入开发阶段前，请确认以下步骤：

- 已参考[快速入门](#)完成产品开通。
- 已明确您的接入方式和采用的[认证方案](#)。
- 已在管理控制台上完成[业务设置](#)。

无线SDK+服务端接入

系统时序图：



适用的认证方案：[RPBasic](#)、[RPManual](#)、[RPBioID](#)、[RPBioOnlyPro](#)、[RPBioOnly](#)、[FVBioOnly](#)

服务端开发参考：

- 接口调用方式：[获取 AccessKey](#)、[API 调用方式](#)
- 相关 API 文档：[发起认证请求\(GetVerifyToken\)](#)、[查询认证状态\(GetStatus\)](#)、[获取认证资料\(GetMaterials\)](#)

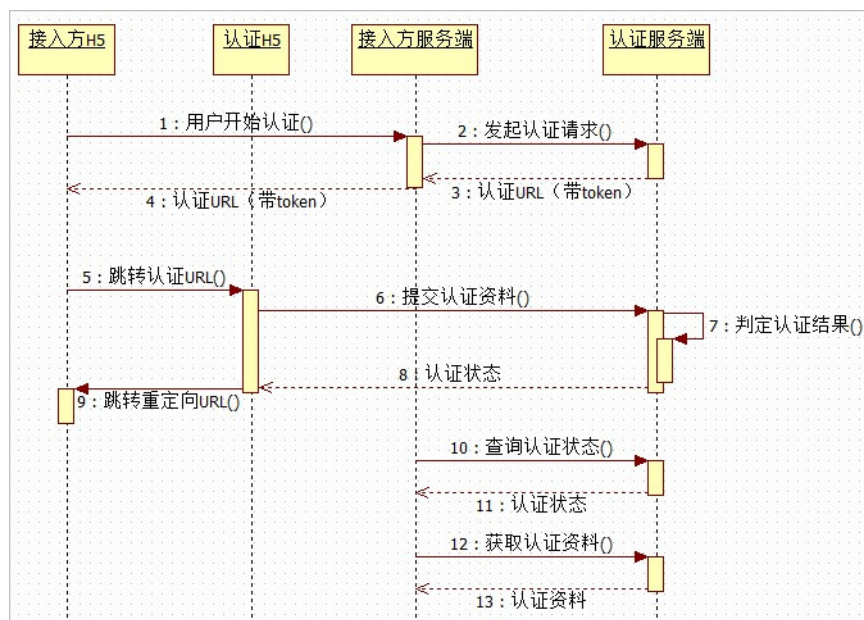
无线客户端开发参考：

- 获取 SDK：[下载无线 SDK](#)

- 集成 SDK：[Android 集成](#)、[iOS 集成](#)

H5+服务端接入

系统时序图：



适用的认证方案：[RPH5BioOnly](#)

服务端开发参考：

- 接口调用方式：[获取 AccessKey](#)、[API 调用方式](#)
- 相关 API 文档：[发起认证请求\(GetVerifyToken\)](#)、[查询认证状态\(GetStatus\)](#)、[获取认证资料\(GetMaterials\)](#)

H5 前端开发参考：

- 认证流程页面入口 URL，从 [发起认证请求\(GetVerifyToken\)](#) 接口中获得
- 认证结果重定向链接：
 - 认证通过跳转指定 URL，可在认证 URL 后加上 successRedirect 参数：`{认证流程页面入口URL}&successRedirect={认证通过时重定向URL}`
 - 认证不通过跳转指定 URL，可在认证 URL 后加上 failRedirect 参数：`{认证流程页面入口URL}&failRedirect={认证不通过时重定向URL}`

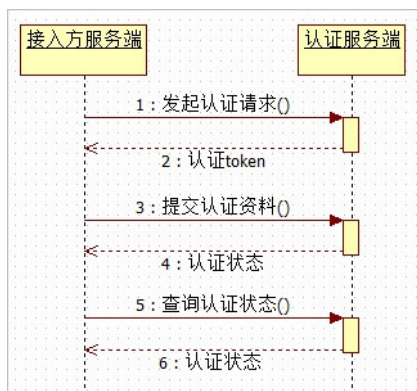


说明：

重定向 URL 需要加上协议头，并先使用 encodeURIComponent 编码后再拼接。

纯服务端接入

系统时序图：



适用的认证方案：[RPMIn](#)

服务端开发参考：

- 接口调用方式：[获取 AccessKey](#)、[API 调用方式](#)
- 相关 API 文档：[发起认证请求\(GetVerifyToken\)](#)、[提交认证资料\(SubmitMaterials\)](#)、[查询认证状态\(GetStatus\)](#)

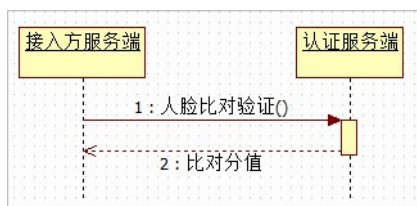
人脸验证



说明：

在正式进入开发阶段前，请确认已参考[快速入门](#)完成产品开通

系统时序图：



服务端开发参考：

- 接口调用方式：[获取 AccessKey](#)、[API 调用方式](#)
- 相关 API 文档：[人脸比对验证\(CompareFaces\)](#)

相关文档

- [API 资源导航](#)
- [API Explorer](#)

- [API错误中心](#)

2 服务端API

2.1 获取AccessKey

介绍如何创建RAM子用户，并授权其访问API和进行控制台操作。

阿里云会对每个访问请求进行身份验证，采用Access Key ID和Access Key Secret进行对称加密的方法来验证请求的发送者身份。其中，Access Key ID和Access Key Secret统称为AccessKey，是阿里云颁发给用户的访问服务所用的密钥。

实人认证支持阿里云账号或其 RAM 子用户的 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，具体操作步骤如下：

1. 创建 RAM 子用户，具体请参考[创建 RAM 用户](#)。
2. 给 RAM 子用户授权系统策略AliyunYundunCloudAuthFullAccess，具体请参考[给 RAM 用户授权](#)。
3. 给 RAM 子用户创建 AccessKey，具体请参考[创建 AccessKey](#)。

2.2 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、签名等等，具体字段请参看下文。
- **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看各个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，具体请参考 API发布说明 。
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考 API 签名机制 中的RPC风格API。
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 API Release Notes

下表记录了历次API更新说明。

API版本号	发布说明
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）。
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：

API版本号	发布说明
	- 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。
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 API说明文档

2.4.1 查询认证状态

描述

接口名：**GetStatus**

查询指定业务场景下一个认证ID的认证状态。

请求方法：支持以HTTPS POST和GET方法发送请求。

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：GetStatus。
RegionId	String	query	是	服务所在地域。取值：cn-hangzhou。
Biz	String	query	是	要查询的业务场景。请先参考 业务设置 在控制台上完成创建。
TicketId	String	query	是	要查询的认证任务ID。通常由业务使用方指定，方便关联业务场景的其他内容。

名称	类型	参数位置	是否必需	描述
				 说明： 需要与当前认证任务在 GetVerifyToken 时的认证ID保持一致。

返回参数

名称	类型	是否必需	描述
StatusCode	Integer	是	认证任务所处的认证状态，取值： -1：未认证。表示没有提交记录。 0：认证中。表示已提交认证，系统正在审核认证资料。 1：认证通过。表示最近一次提交的认证资料已通过审核，当前认证任务完结。 2：认证不通过。表示最近一次提交的认证资料未通过审核，当前认证任务还可以继续发起提交。
SimilarityScore	Float	是	认证过程中所提交的人脸照片和身份证上的头像的相似程度分值。取值范围为[0,100]，分数越大相似度越高。  说明： 只有提交的认证资料中同时包含人脸照片和身份证人像面照片，该值才有意义。
AuditConclusions	String	是	认证状态为“认证不通过”时的原因描述。不通过原因包括但不限于以下几种情况： - 身份证照片模糊，光线问题造成字体无法识别。 - 身份证照片信息与认证提交信息不一致。 - 提交的图片非身份证照片。建议您请按引导提交本人有效身份证照片。

错误码

错误代码	英文描述	中文描述	HTTP 状态码
Error.InternalError	The request processing has failed	服务内部出现未知错误，请联系技术支持排查。	500

错误代码	英文描述	中文描述	HTTP 状态码
	due to some unknown error.		
InvalidParam. ParamMissing	Please check if there is param empty in your input.	参数错误，请检查参数是否为空。	400
Error.NoPermissionAccess	No permission to access our service.	没有权限访问服务，请确认是否服务已开通。	403
InvalidParam. InvalidBiz	The biz is not valid.	参数 Biz 不合法，请与控制台上创建的保持一致。	400

示例

关于使用**SDK**开发包的示例，请参考以下具体SDK 开发包使用文档中的认证方案说明：[Java](#)、[PHP](#)、[Python](#)、[.NET](#)、[Node.js](#)、[Go](#)。

以下是拼接**HTTPS**请求的相关示例：

请求示例

```
https://cloudauth.aliyuncs.com/?Action=GetStatus
&RegionId=cn-hangzhou
&Biz=RPBasic
&TicketId=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'?>
<GetStatusResponse>
  <Data>
    <StatusCode>1</StatusCode>
    <SimilarityScore>93.12</SimilarityScore>
    <AuditConclusions>不通过原因</AuditConclusions>
  </Data>
  <Success>true</Success>
  <Code>1</Code>
</GetStatusResponse>
```

- JSON格式

```
{
  "Data": {
    "SimilarityScore": 93.12,
    "StatusCode": 1,
```

```
"AuditConclusions": "不通过原因"
},
"Success": true
}
```

2.4.2 发起认证请求

描述

接口名：**GetVerifyToken**

请求方法：支持以HTTPS POST和GET方法发送请求。

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。

关于上传图片地址的说明

在传入图片时，您需要上传其对应的HTTP/OSS地址或base64编码。

- HTTP地址：可访问的公网HTTP地址。例如，`http://image-demo.img-cn-hangzhou.aliyuncs.com/example.jpg`。
- OSS地址：使用STS功能上传图片到实名认证服务指定的OSS bucket后，所生成的OSS文件地址，格式为`oss://<STSToken.bucketName>:<path_to_file>`。[查看详情](#)。
- base64编码：通过base64编码的图片，格式为`base64://<图片 base64 字符串>`。



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。
- 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。
- 使用base64传递图片时，接口的请求方法需要改成 POST；图片base64字符串需要去掉头部描述，如`data:image/png;base64,`。

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：GetVerifyToken。
RegionId	String	query	是	服务所在地域。取值：cn-hangzhou。

名称	类型	参数位置	是否必需	描述
Biz	String	query	是	使用实人认证服务的业务场景。请先参考 业务设置 在控制台完成创建。
TicketId	String	query	是	标识一次认证任务的唯一ID。通常由业务使用方指定，方便关联业务场景的其他内容。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认证 ID。
Binding	String	body	否	认证扩展材料（JSON 格式）。具体内容请参考 扩展参数Binding说明 。
UserData	String	query	否	业务数据（JSON 格式），通常用来传递业务上下文内容。

扩展参数Binding说明

部分场景中，业务方在发起认证请求前已经收集了一部分认证资料，需要传递给实人认证服务，这种情况下就需要使用**Binding**扩展参数。目前支持**Binding**的认证资料包括：



说明：

具体传入哪些字段，视您选择的认证方案和实际业务需求而定，具体请参考[认证方案](#)中的说明。

- **Name**：姓名。字符串格式，例如，李三。
- **IdentificationNumber**：身份证号。字符串格式，例如，630546198009090099。
- **IdCardFrontPic**：身份证人像面照片。字符串格式，具体请参考[关于上传图片地址的说明](#)。
- **IdCardBackPic**：身份证国徽面照片。字符串格式，具体请参考[关于上传图片地址的说明](#)。
- **FacePic**：人像照。字符串格式，具体请参考[关于上传图片地址的说明](#)。
- **IDPic**：二代身份证芯片中存储的头像照片。字符串格式，具体请参考[关于上传图片地址的说明](#)。
- **FaceRetainedPic**：人像留底照片。字符串格式，具体请参考[关于上传图片地址的说明](#)。



注意：

具体传入哪些字段，视您选择的认证方案和实际业务需求而定，具体请查看[认证方案](#)中的说明。

应用示例

业务方在自己的业务流程中已经采集了用户的姓名、身份证号、身份证人像面照片、身份证国徽面照片，而希望使用实人认证服务对用户进行活体检测并确认是本人操作。在这种情况下，就需要在发起认证请求时使用**Binding**参数指定这些采集到的数据。该场景下，**Binding**的内容示例如下：

```
{
  \"Name\": \"李三\",
  \"IdentificationNumber\": \"584129192830192233\",
  \"IdCardFrontPic\": \"http://image-demo.img-cn-hangzhou.aliyuncs.com/example.jpg\",
  \"IdCardBackPic\": \"base64:///9j/4AAQSkZJRgABAQEASABIAAD/2...\"
}
```

返回参数

名称	类型	是否必需	描述
VerifyToken	Map	是	该次认证的Token，用来串联认证请求中的各个接口。具体结构描述见 VerifyToken 。
StsToken	Map	否	如果业务方有额外的文件需要一并提交认证服务，可以使用 STSToken 将其上传到认证服务指定的地方。操作方法参考 使用STSToken 上传文件示例 。 STSToken 的具体结构描述见 STSToken 。
CloudauthPageUrl	String	否	认证流程页面入口URL。  说明： H5+服务端接入的认证方案需要使用此字段，用作前端 H5 跳转到认证流程的 URL。

表 2-1: VerifyToken

名称	类型	是否必需	描述
Token	String	是	一次认证会话的标识。
DurationSeconds	Integer	是	认证会话超时时间，单位为秒。一般为1,800s。

表 2-2: STSToken

名称	类型	是否必需	描述
AccessKeyId	String	是	OSS AccessKey ID。
AccessKeySecret	String	是	OSS AccessKey密钥。
Expiration	String	是	STS token过期时间。
EndPoint	String	是	OSS endpoint。
BucketName	String	是	OSS bucket，认证服务指定的文件上传 Bucket。
Path	String	是	一个认证请求生成的用来上传的目录，业务方需要将文件上传到这个目录。
Token	String	是	STS 访问Token。

使用STSToken上传文件示例

为了让业务方能够将图片上传到实人认证服务指定的OSS Bucket中，需要使用OSS提供的安全凭证管理（Security Token Service，简称 STS）功能。STS 能够依据bucket owner指定的访问策略生成一个临时的STS token给业务方，作为临时上传文件使用的凭据。关于STS的更多信息，请参考[OSS访问控制](#)。

1. 使用OSS SDK上传图片。

例如，有一张人脸图片名字为 face.jpg，需要上传到实人认证服务进行比对，可以参考以下流程（Java 语言）：

```
String bucketName = uploadToken.bucketName;
String endpoint = uploadToken.endpoint;
String accessKeyId = uploadToken.accessKeyId;
String accessKeySecret = uploadToken.accessSecret;
String token = uploadToken.token;
// 初始化客户端
OSSClient ossClient = new OSSClient(endpoint, accessKeyId,
accessKeySecret, token);
String fileKey = uploadToken.path + "face.jpg"; // c3fef70f36
ca4b0d8203a499beaa9f80/face.jpg
byte[] content = getBytesOf("face.jpg");
// 上传图片
ossClient.putObject(bucketName, fileKey, content);
// 关闭client
ossClient.shutdown();
```

2. 使用[SubmitMaterials](#)接口上传 OSS 文件给实人认证服务。

在与 OSS 对接成功上传图片后，获取到图片的文件路径（例如，c3fef70f36ca4b0d8203a499beaa9f80/face.jpg）。然后，需要将此图片的文件路径地址告知实人认证服务端。

对于已成功上传至 OSS 的图片，传递给实人认证服务端的格式应为oss://<uploadToken.bucketName>:c3fef70f36ca4b0d8203a499beaa9f80/face.jpg。

将此完整的图片文件名通过[SubmitMaterials](#)接口提交给实人认证服务即可。

错误码

错误代码	英文描述	中文描述	HTTP 状态码
Error.InternalError	The request processing has failed due to some unknown error.	服务内部出现未知错误，请联系技术支持排查。	500
InvalidParam. ParamMissing	Please check if there is param empty in your input	参数错误，请检查参数是否为空。	400
Error.NoPermissionAccess	No permission to access our service	没有权限访问服务，请确认是否服务已开通。	403
InvalidParam. InvalidBiz	The biz is not valid	参数Biz不合法，请与控制台上创建的保持一致。	400

示例

关于使用**SDK**开发包的示例，请参考以下具体SDK 开发包使用文档中的认证方案说明：[Java](#)、[PHP](#)、[Python](#)、[.NET](#)、[Node.js](#)、[Go](#)。

以下是拼接**HTTPS**请求的相关示例：

请求示例

```
https://cloudauth.aliyuncs.com/?Action=GetVerifyToken
&RegionId=cn-hangzhou
&Biz=RPBasic
&TicketId=39ecf51e-2f81-4dc5-90ee-ff86125be683
&UserData=%7B%22orderId%22%3A+%221391012388%22%7D
&<公共请求参数>
```

```
请求body
Binding=%7B%22Name%22%3A+%22%E6%9D%8E%E4%B8%89%22%7D
```

返回示例

- XML格式

```
<?xml version='1.0' encoding='UTF-8'?>
<GetVerifyTokenResponse>
  <Data>
    <StsToken>
      <AccessKeySecret>OSSAccessKeySecret</AccessKeySecret>
      <BucketName>verify-external-image</BucketName>
      <AccessKeyId>OSSAccessKeyId</AccessKeyId>
      <Expiration>2017-09-14T02:28:20Z</Expiration>
      <EndPoint>https://oss-cn-shanghai.aliyuncs.com</EndPoint>
    >
    <Token> CAIS2QJ1q6Ft5B2yfSjIpoJNLPyNqehJ4I7dUWLP
tzMuFs1qvaDDujz2IHpFenBqBuActP8xmWFW5v0Slo15QZFLHhecNZop4pkL
oQj8M9fL5Zfq4+Rf0sb6EDafAUZiUXtL7KarIunGc9KBNo2/00amMkGHFf
PdlihNoLzxvWDR98LXw6+QCNbDdNNXGVLo9MbMn2jUpTINQXx0FLNEG1iuAd3
lRl1i8KFz9ab9wDVgXDj1+YRvP6RGJW/aNR2N5oNfbWZ1edtJK3ay3wSuVoY6q
psjaJVvDrXotyHBVBVphSD0uXZtddlIgeiPfBiSvdIoaSsnqAo5+aNzNv5k00X
Zr5cDSjVA8KcuJKeQ7P1aI9nKOigaimSgoC1W80r419+UxUyLxhXftctEHh0BCE3
RyvSQq3dowqbPF75FPjfivpojMsrnw6457uQLl3KXK+YzSsI3he3n9aLxnQagAG
uwJ70RAZMwZm0UqNZ9uICFM2L39GX6/H22KDq2BYG04bMcS49V+CuBuzTK1ME
gRE5ELfBq6ttk4/Arah6gUomug6Yi8ZZCl0Zghgl36j5FEHeOYMS/RGnyiLVhRb+
biHoncq/PsD/m8bv3VVQIcKQQ9gF71+WzebokSmhSw3A7w==
    </Token>
    <Path>bbff330091de48c585b637f03237a6db</Path>
  </StsToken>
  <VerifyToken>
    <DurationSeconds>1800</DurationSeconds>
    <Token>bbff330091de48c585b637f03237a6db</Token>
  </VerifyToken>
  <CloudauthPageUrl>https://h5.m.taobao.com/user_verify/
cloudauth.html?token=1234567890abcdefg</CloudauthPageUrl>
</Data>
<Success>true</Success>
<Code>1</Code>
</GetVerifyTokenResponse>
```

- JSON格式

```
{
  "Code": "1",
  "Data": {
    "StsToken": {
      "AccessKeyId": "OSSAccessKeyId",
      "AccessKeySecret": "OSSAccessKeySecret",
      "BucketName": "verify-external-image",
      "EndPoint": "https://oss-cn-shanghai.aliyuncs.com",
      "Expiration": "2017-09-09T13:01:32Z",
      "Path": "92790974cd384f29a949b47f1c4ca3a5/",
      "Token": "CAIS2QJ1q6Ft5B2yfSjIpKaGDuzDqJpP5vWfNhLY1mMPPfxphv
Pa1Dz2IHpFenBqBuActP8xmWFW5v0SlokiEcaUHR2bMcB8tcQL/1nwatud6sb
psrJdg8GuED0aVkJZ0DUsP7aarIunGc9KBNo2/00amMkGHFfPdlihNoLzxvWDR98LXw6
+QCNbDdNNXGVLo9MbMn2jUpTINQXx0FLNEG1iuAd3lRl1i8KFz9ab9wDVgXDj
1+YRvP6RGJW/aNR2N5oNfbWZ1edtJK3ay3wSuVoY6qpsjaJVvDrXotyHBVBVph
SD0uXZtYw1c14hN/dmEKIf/KTyZ6xx67CCndioxU9GYepcCC2CA8KcuJKeQ7P1a
I9nKOigaimSgoC1W80r419+UxUyLxhXftctEHh0BCE3RyvSQq3dowqbPF75FPjf
ivpojMsrnw6457uQLl3KXK+YzSsI3he3n9aLxnQagAGferYB1UMADL0Lyw3Bd91
xmtPTfJD8QszrYAladv2D7CxsqP7jdBoJ6gRQoE9U0NEibHGjFeNZ3C8VN5kmnU+
aY6lnwgcZ+7nEXBnrh4hl56ubNXb8Tkx5LDEbfXrZmnfv5LbuUn+790kvQPQvfmL3xM+
7ciWyAzmeBBGTIdf8w=="
    },
  },
}
```

```
"VerifyToken": {
  "Expire": 1800,
  "Token": "92790974cd384f29a949b47f1c4ca3a5"
},
"CloudauthPageUrl": "https://h5.m.taobao.com/user_verify/cloudauth.html?token=1234567890abcdefg"
},
"Success": true
}
```

2.4.3 提交认证资料

描述

接口名：**SubmitMaterials**

请求方法：仅支持以HTTPS POST方法发送请求。

服务端提交认证材料到实人认证服务进行审核，结果同步返回。

通常**SubmitMaterials**接口会同步返回认证结果。但是如果出现内部服务超时等现象，可能无法实时获取到认证的最终状态，此时接口会返回“认证中”状态。您在开发接入设计中需要考虑到这种情况，并可以使用[GetStatus](#)接口轮询最新的状态。

关于上传图片地址的说明

在传入图片时，您需要上传其对应的HTTP/OSS地址或base64编码。

- HTTP地址：可访问的公网HTTP地址。例如，`http://image-demo.img-cn-hangzhou.aliyuncs.com/example.jpg`。
- OSS地址：使用STS功能上传图片到实人认证服务指定的OSS bucket后，所生成的OSS文件地址，格式为`oss://<STSToken.bucketName>:<path_to_file>`。 [查看详情](#)。
- base64编码：通过base64编码的图片，格式为`base64://<图片 base64 字符串>`。



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。
- 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。
- 使用base64传递图片时，接口的请求方法需要改成 POST；图片base64字符串需要去掉头部描述，如`data:image/png;base64,`。

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：SubmitMaterials。
RegionId	String	query	是	服务所在地域。取值：cn-hangzhou。
VerifyToken	String	body	是	认证会话标识，通过 GetVerifyToken 接口获取。
Material.n.MaterialType	材料类型	body	是	提交审核的认证材料的类型。取值：  说明： 需要传入哪些字段取决于您接入时确定的认证方案，具体见认证方案中说明。 • Name：姓名。对应材料值为字符串格式，例如，张三。 • IdentificationNumber：身份证号。对应材料值为字符串格式，例如，630546198009090099。 • FacePic：人像照。对应材料值为字符串格式，具体请参考关于上传图片地址的说明。 • IdCardFrontPic：身份证人像面照片。对应材料值为字符串格式，具体请参考关于上传图片地址的说明。 • IdCardBackPic：身份证国徽面照片。对应材料值为字符串格式，具体请参考关于上传图片地址的说明。 • IDPic：二代身份证芯片中存储的头像照片。对应材料值为字符串格式，具体请参考关于上传图片地址的说明。 • Mobile：手机号（该手机号用于辅助风险判断，而非与手机运营商

名称	类型	参数位置	是否必需	描述
				实名信息进行核验。) 对应取值为字符串格式，例如，13983822930。 n为整型可变参数，当需要传入多种类型的材料时，依次通过Material.1.MaterialTy、Material.2.MaterialTy、...、Material.10.MaterialTy传入类型值。n最大为10，且必须和Material.n.Value中的n匹配。
Material.n.Value	材料值	body	是	提交审核的认证材料的取值，与具体的材料类型对应，具体见Material.n.MaterialType描述。 n为整型可变参数，最大为10，且必须和Material.n.MaterialType中的n匹配。

返回参数

名称	类型	是否必需	描述
VerifyStatus	Map	是	提交认证资料后的认证状态。具体结构描述见 VerifyStatus 。

表 2-3: VerifyStatus

名称	类型	是否必需	描述
StatusCode	Integer	是	一个认证任务所处的认证状态。取值： -1：未认证。表示没有提交记录。 0：认证中。表示已提交认证，系统正在审核认证资料。 1：认证通过。表示最近一次提交的认证资料已通过审核，当前认证任务完结。 2：认证不通过。表示最近一次提交的认证资料未通过审核，当前认证任务还可以继续发起提交。

名称	类型	是否必需	描述
Similarity Score	Float	是	认证过程中所提交的人脸照片和身份证上的头像的相似程度分值。取值范围为[0,100]，分数越大相似度越高。  说明： 只有提交的认证资料中同时包含人脸照片和身份证人像面照片，该值才有意义。
AuditConclusions	String	是	认证状态为“认证不通过”时的原因描述。不通过原因包括但不限于以下几种情况： • 身份证照片模糊，光线问题造成字体无法识别。 • 身份证照片信息与认证提交信息不一致。 • 提交的照片非身份证照片。 建议您请按引导提交本人有效身份证照片。

错误码

错误代码	英文描述	中文描述	HTTP状态码
Error.InternalError	The request processing has failed due to some unknown error.	服务内部出现未知错误，请联系技术支持排查。	500
InvalidParam. ParamMissing	Please check if there is param empty in your input.	参数错误，请检查参数是否为空。	403
Error.NoPermissionAccess	No permission to access our service.	没有权限访问服务，请确认是否服务已开通。	403
InvalidParam. VerifyTokenExpired	The verify token was expired.	VerifyToken 过期。VerifyToken 是有时效性的，请确保在 token 失效前进行提交。	400
InvalidParam. MaterialsEmpty	The materials list is empty.	Materials 列表为空。	400
InvalidParam. MaterialsNotValid	The param materials is not valid.	请确保提交的材料类型正确性。	403

错误代码	英文描述	中文描述	HTTP状态码
Error.DuplicatedTicketId	A record associated with your ticketId has existed.	重复提交。一个 TicketId 参数已经提交，且认证状态为认证中或认证通过时，再次提交会出现此错误。	403
Error.SubmitMaterialsError	SubmitMaterials action failed due to internal error.	提交处理失败。通常是实人认证服务内部处理异常，请联系技术支持排查。	500
Error.MaterialsUploadError	The materials upload failed.	SubmitMaterials 在上传认证资料中出现错误，请检查图片地址格式正确，图片大小未超限，使用 base64 传图的已设置请求方法为POST。	500
Error.MaterialsInsufficient	The number of materials does not meet the requirements for verification.	提交认证的材料种类不满足要求，请检查上传的材料种类是否符合业务场景要求。	403

示例

关于使用**SDK**开发包的示例，请参考以下具体SDK 开发包使用文档中的认证方案说明：[Java](#)、[PHP](#)、[Python](#)、[.NET](#)、[Node.js](#)、[Go](#)。

以下是拼接**HTTPS**请求的相关示例：

请求示例

```
https://cloudauth.aliyuncs.com/?Action=SubmitMaterials
&RegionId=cn-hangzhou
&<[公共请求参数]>
```

请求body

```
VerifyToken=8eb57ce35d584cbb9724198cb0ac6b60
&Material.1.MaterialType=FacePic
&Material.1.Value=http%3A%2F%2Fjiangsu.china.com.cn%2Fuploadfile%
2F2015%2F0114%2F1421221304095989.jpg
&Material.2.MaterialType=IDPic&SignatureNonce=c71f14e8-f2e7-45fd-9c5a-
721cf6b38c4a
&Material.2.Value=oss%3A%2F%2Fverify-img%3A715559d79b8d49c4aa
bfdad3f6a407740SS.JPG
```

```
&Material.3.MaterialType=IdCardFrontPic&Signature=azM4ZgJ7auesjlWQuMBF9X2Tnc8%3D
&Material.3.Value=oss%3A%2F%2Fverify-img%3A715559d79b8d49c4aabfdad3f6a407740SS.JPG
&Material.4.MaterialType=IdCardBackPic
&Material.4.Value=oss%3A%2F%2Fverify-img%3A715559d79b8d49c4aabfdad3f6a407740SS.JPG
&Material.5.MaterialType=IdentificationNumber
&Material.5.Value=630546198009090099
&Material.6.MaterialType=Name
&Material.6.Value=%E6%9D%8E%E4%B8%89
```

返回示例

- XML格式

```
<?xml version='1.0' encoding='UTF-8'?>
<SubmitMaterialsResponse>
  <Data>
    <VerifyStatus>
      <StatusCode>1</StatusCode>
      <SimilarityScore>93.12</SimilarityScore>
      <AuditConclusions>不通过原因</AuditConclusions>
    </VerifyStatus>
  </Data>
  <Success>true</Success>
  <Code>1</Code>
</SubmitMaterialsResponse>
```

- JSON格式

```
{
  "Code": "1",
  "Data": {
    "VerifyStatus": {
      "MaterialScore": 95.22905,
      "StatusCode": 1
      "AuditConclusions": "不通过原因"
    }
  },
  "Success": true
}
```

2.4.4 获取认证资料

描述

接口名：**GetMaterials**

请求方法：支持以HTTPS POST和GET方法发送请求。

收到认证请求后，业务系统通过该接口获取用户上传的资料（人脸图、证件图）及算法识别的资料（OCR的地址等）。

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：GetMaterials。
RegionId	String	query	是	服务所在地域。取值：cn-hangzhou。
Biz	String	query	是	使用实人认证服务的业务场景。请先参考 业务设置 在控制台完成创建。
TicketId	String	query	是	标识一次认证任务的唯一ID。通常由业务使用方指定，方便关联业务场景的其他内容。  说明： 需要与当前认证任务在GetVerifyToken中指定的TicketId保持一致。

返回参数

名称	类型	是否必需	描述
Name	String	是	姓名。
IdentificationNumber	String	是	证件号。
IdCardType	String	是	证件类型。identityCard代表身份证。
IdCardStartDate	String	否	证件有效期起始日期。格式：yyyyMMdd。依赖于算法识别，非必返回字段，可能存在无法识别的情况。
IdCardExpiry	String	否	证件有效期截止日期。格式：yyyyMMdd。依赖于算法识别，非必返回字段，可能存在无法识别的情况。
Address	String	否	证件地址。通过OCR算法识别出的地址（JSON 数据格式），结构描述见 Address 参数示例 。 证件地址依赖于算法识别，非必返回字段，可能存在无法识别或识别不全的情况。
Sex	String	否	证件性别。返回值的集合为{m, f}，其中m代表男性，f代表女性。

名称	类型	是否必需	描述
IdCardFrontPic	String	否	证件照正面图片HTTP地址（调用本接口后1小时内可访问）。若证件类型为身份证，则为身份证人像面照片。
IdCardBackPic	String	否	证件照背面图片HTTP地址（调用本接口后1小时内可访问）。若证件类型为身份证，则为身份证国徽面照片。
FacePic	String	是	认证过程中拍摄的人像正面照图片HTTP地址（调用本接口后1小时内可访问）。
EthnicGroup	String	否	证件上的民族。依赖于算法识别，非必返回字段，可能存在无法识别的情况。

Address示例



说明：

以下地址通过身份证OCR识别获得，可能存在识别错误或不全的情况。其中，**province**为省/直辖市，**city**为市，**area**为区/县/县级市，**town**为街道/镇，**detail**为详细地址。由于街道/镇一级的行政划分经常匹配不到，**town**的返回值为0。

```
{
  "address": {
    "area": {
      "text": "余杭区",
      "value": "330110",
      "key": "area"
    },
    "province": {
      "text": "浙江省",
      "value": "330000",
      "key": "province"
    },
    "town": {
      "value": "0",
      "key": "town"
    },
    "city": {
      "text": "杭州市",
      "value": "330100",
      "key": "city"
    }
  },
  "detail": "五常街道永福社区文一西路977号"
```

```
}

```

错误码

错误代码	英文描述	中文描述	HTTP 状态码
Error.InternalError	The request processing has failed due to some unknown error.	服务内部出现未知错误，请联系技术支持排查。	500
InvalidParam. ParamMissing	Please check if there is param empty in your input.	参数错误，请检查参数是否为空。	400
Error.NoPermissionAccess	No permission to access our service.	没有权限访问服务，请确认是否服务已开通。	403
InvalidParam. InvalidBiz	The biz is not valid	参数 Biz 不合法，请与控制台上创建的保持一致。	400
InvalidParam. RecordNotFound	There is not record associated with your ticketId.	没有找到和 TicketId 关联的记录。	404
Error.GetMaterialsIsError	GetMaterials action failed due to internal error.	查询认证资料过程中出现内部未知错误，请联系技术支持排查。	500

示例

关于使用**SDK**开发包的示例，请参考以下具体SDK 开发包使用文档中的认证方案说明：[Java](#)、[PHP](#)、[Python](#)、[.NET](#)、[Node.js](#)、[Go](#)。

以下是拼接**HTTPS**请求的相关示例：

请求示例

```
https://cloudauth.aliyuncs.com/?Action=GetMaterials
&Biz=RPBasic
&TicketId=1839171847
&<[公共请求参数]>
```

返回示例

- XML格式

```
<?xml version='1.0' encoding='UTF-8'?>
```

```
<GetMaterialsResponse>
  <Data>
    <Name>李明</Name>
    <FacePic><![CDATA[http://image-demo.img-cn-hangzhou.aliyuncs.com/example.jpg]]></FacePic>
    <IdCardType>identityCard</IdCardType>
    <Address>{ "detail": "2号大街18号", "address": { "area": { "text": "江干区", "key": "area" }, "province": { "text": "浙江省", "key": "province" }, "city": { "text": "杭州市", "key": "city" } } }</Address>
    <IdCardBackPic>
      <![CDATA[http://verify-img.cn-shanghai.img.aliyun-inc.com/542fd010b3b04ef18193a94d8739cf970SS.JPG?Expires=1505348837&OSSAccessKeyId=IJ95qE4nJQY6t6Lk&Signature=WeIU6zzeXXpBhwf7uwig0p7I7Hk%3D]]></IdCardBackPic>
    <IdCardFrontPic>
      <![CDATA[http://verify-img.cn-shanghai.img.aliyun-inc.com/715559d79b8d49c4aabfdad3f6a407740SS.JPG?Expires=1505348836&OSSAccessKeyId=IJ95qE4nJQY6t6Lk&Signature=R9bwCimlsXLGYaIZ8S%2BppxNAZJ0%3D]]></IdCardFrontPic>
    <IdentificationNumber>629212199002108316</IdentificationNumber>
    <IdCardStartDate>19900210</IdCardStartDate>
    <IdCardExpiry>19900210</IdCardExpiry>
    <Sex>m</Sex>
    <EthnicGroup>汉</EthnicGroup>
  </Data>
  <Success>true</Success>
  <Code>1</Code>
</GetMaterialsResponse>
```

- JSON格式

```
{
  "Code": "1",
  "Data": {
    "Address": "{ \"detail\": \"2号大街1号\", \"address\": { \"area\": { \"text\": \"江干区\", \"value\": \"330104\", \"key\": \"area\" }, \"province\": { \"text\": \"浙江省\", \"value\": \"330000\", \"key\": \"province\" }, \"city\": { \"text\": \"杭州市\", \"value\": \"330100\", \"key\": \"city\" } } }",
    "FacePic": "http://image-demo.img-cn-hangzhou.aliyuncs.com/example.jpg",
    "IdCardBackPic": "http://verify-img.cn-shanghai.img.aliyun-inc.com/581fd010b3b04ef18193a94d8739cf070SS.JPG?Expires=1505301155&OSSAccessKeyId=IJ95qE4nJQY6t6Lk&Signature=nYEZoBcdBro4Yfr4QQSv%2B3/rM24%3D",
    "IdCardStartDate": "19900210",
    "IdCardExpiry": "19900210",
    "IdCardFrontPic": "http://verify-img.cn-shanghai.img.aliyun-inc.com/915559f79b8d49c8aabfdad3f6a407740SS.JPG?Expires=1505301155&OSSAccessKeyId=IJ95qE4nJQY6t6Lk&Signature=jgW4rqB50STbDDo1Xlf7rwL9N80%3D",
    "IdCardType": "identityCard",
    "IdentificationNumber": "545230199002108328",
    "Name": "李明",
    "Sex": "m",
    "EthnicGroup": "汉"
  },
}
```

```
"Success": true
}
```

2.4.5 人脸比对验证

描述

接口名：**CompareFaces**

请求方法：仅支持以HTTPS POST方法发送请求。

指定两张人脸图片进行比对，输出两张图片中人脸的相似度分值作为结果。

- 指定的比对图片中至少应含有一个图片类型是人脸照（FacePic）。
- 一张图片中含有多个人脸时，算法会自动选取图片中占幅最大的人脸。
- 两张比对的图片中，如果其中一张检测不到人脸，系统会返回“未检测到人脸”的错误提示。

关于上传图片地址的说明

在传入图片时，您需要上传图片对应的HTTP/OSS地址或base64编码。

- HTTP地址：可访问的公网HTTP地址。例如，`http://image-demo.img-cn-hangzhou.aliyuncs.com/example.jpg`。
- OSS地址：使用STS功能上传图片到实人认证服务指定的OSS bucket后，所生成的OSS文件地址，格式为`oss://<STSToken.bucketName>:<path_to_file>`。[查看详情](#)。
- base64编码：通过base64编码的图片，格式为`base64://<图片 base64 字符串>`。



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。
- 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。
- 使用base64传递图片时，接口的请求方法需要改成 POST；图片base64字符串需要去掉头部描述，如`data:image/png;base64,`。

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：CompareFaces。

名称	类型	参数位置	是否必需	描述
RegionId	String	query	是	服务所在地域。取值：cn-hangzhou。
SourceImageType	String	body	是	图片1的类型，取值： - FacePic：用户人脸照 - IDPic：用户二代身份证芯片中的头像照（通常由二代身份证阅读仪设备读取并解码获得）
SourceImageValue	String	body	是	图片1的地址。具体请参考 关于上传图片地址的说明 。
TargetImageType	String	body	是	图片2的类型，取值： - FacePic：用户人脸照 - IDPic：用户二代身份证芯片中的头像照（通常由二代身份证阅读仪设备读取并解码获得）
TargetImageValue	String	body	是	图片2的地址。具体请参考 关于上传图片地址的说明 。

返回参数

名称	类型	是否必需	描述
Similarity Score	Float	是	两个图片中人脸的相似程度。取值范围为[0, 100]，值越大相似度越高。
Confidence Thresholds	String	是	人脸比对置信度阈值。返回内容是JSON Object，具体结构为"key": "value"。其中， - key是误识率，即将其他人误识别为指定人员的概率。 - value是对应的阈值。

错误码

错误代码	英文描述	中文描述	HTTP状态码
Error.InternalError	The request processing has failed	服务内部出现未知错误，请联系技术支持排查。	500

错误代码	英文描述	中文描述	HTTP状态码
	due to some unknown error.		
InvalidParam. ParamMissing	Please check if there is param empty in your input.	参数错误，请检查参数是否为空。	403
Error.NoPermissionAccess	No permission to access our service.	没有权限访问服务，请确认是否服务已开通。	403
Error.CompareFacesError	CompareFaces action failed due to internal error.	人脸比对过程中出现内部未知错误，请联系技术支持排查。	500
InvalidParam. ImageTypeEmpty	The image type is empty.	SourceImageType 或者 TargetImageType 字段为空。	400
InvalidParam. FacePicNotGiven	At least one of the two given images has a type of FacePic.	进行比对的两个图片中，至少其中一个图片的类型应该设置为人脸照FacePic。	400
InvalidParam. ImageValuePatternError	The pattern of value of given image is not acceptable.	图片地址格式错误，请检查 ImageValue 是否符合图片地址格式的要求。	400
InvalidParam. ImageTypeInvalid	The image type is not valid.	SourceImageType / TargetImageType 取值不为 FacePic 或者 IDPic。	400
Error.NoFaceDetected	No face detected from given images.	指定图片中没有检测到人脸。	400

示例

关于使用**SDK**开发包的示例，请参考以下具体SDK 开发包使用文档中的认证方案说明：[Java](#)、[PHP](#)、[Python](#)、[.NET](#)、[Node.js](#)、[Go](#)。

以下是拼接**HTTPS**请求的相关示例：

请求示例

```
https://cloudauth.aliyuncs.com/?Action=CompareFaces
&RegionId=cn-hangzhou
&<公共请求参数>

请求body
SourceImageType=FacePic
&SourceImageValue=http%3A%2F%2Fjiangsu.china.com.cn%2Fuploadfile%
2F2015%2F0114%2F1421221304095989.jpg
&TargetImageType=IDPic
&TargetImageValue=http%3A%2F%2Fjiangsu.china.com.cn%2Fuploadfile%
2F2015%2F0114%2F1421221304095989.jpg
```

返回示例

- XML格式

```
<?xml version='1.0' encoding='UTF-8'?>
<CompareFacesResponse>
  <Data>
    <ConfidenceThresholds>{"0.0001":"90.07","0.001":"80.02","0.
01":"70.01"}</ConfidenceThresholds>
    <SimilarityScore>0.9825757</SimilarityScore>
  </Data>
  <Success>true</Success>
  <Code>1</Code>
</CompareFacesResponse>
```

- JSON格式

```
{
  "Code": "1",
  "Data": {
    "ConfidenceThresholds": "{\"0.0001\":\"90.07\", \"0.001\":\"80.01\", \"0.01\":\"70.02\"}",
    "SimilarityScore": 98.7913
  },
  "Success": true
}
```



说明：

关于示例中的人脸比对置信度阈值（**confidenceThresholds**）：

- "0.0001": "90.07"表示误识率为0.01%时的对应阈值为90.07。
- "0.001": "80.01"表示误识率为0.1%时的对应阈值为80.01。
- "0.01": "70.02"表示误识率为1%时的对应阈值为70.02。

置信度阈值会根据不同图像和算法动态给出，请不要将置信度阈值持久化。

2.4.6 人脸有效性检测

描述

接口名：**DetectFaceAttributes**

请求方法：仅支持以HTTPS POST方法发送请求。

对输入照片中人脸的有效性相关属性进行检测，便于业务方判定该照片是否满足自身业务留存或比对的要求。目前支持的人脸有效性相关属性有：是否人脸、是否模糊、是否戴眼镜、脸部姿态、性别、年龄、是否笑脸等。

关于上传图片地址的说明

在传入图片时，您需要上传其对应的HTTP/OSS地址或base64编码。

- HTTP地址：可访问的公网HTTP地址。例如，`http://image-demo.img-cn-hangzhou.aliyuncs.com/example.jpg`。
- OSS地址：使用STS功能上传图片到实名认证服务指定的OSS bucket后，所生成的OSS文件地址，格式为`oss://<STSToken.bucketName>:<path_to_file>`。[查看详情](#)。
- base64编码：通过base64编码的图片，格式为`base64://<图片 base64 字符串>`。



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。
- 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。
- 使用base64传递图片时，接口的请求方法需要改成 POST；图片base64字符串需要去掉头部描述，如`data:image/png;base64,`。

请求参数

名称	类型	参数位置	是否必需	描述
MaterialValue	String	body	是	待检测照片，格式描述见 关于上传图片地址的说明 。
RetAttributes	String	body	是	指定需要检测的内容，检测内容越多接口耗时越大。取值： • Facetype：是否人脸 • Headpose：人脸姿态

名称	类型	参数位置	是否必需	描述
				- Glasses：是否戴眼镜 - Blur：是否模糊 - Smiling：是否笑脸 - Gender：性别 - Age：年龄
MaxFaceNum	Integer	body	是	最多检测多少张人脸，默认传1。
DontSaveDB	Boolean	body	是	照片是否不入库，默认传true。
ClientTag	String	body	是	照片标签，默认传null。
MaxNumPhotosPerCategory	Integer	body	是	如果照片需要入库，相同ClientTag 保留的最大张数，新的照片会替换最老的，默认传1。

返回参数

名称	类型	是否必需	描述
ImgWidth	Integer	是	原图宽度，单位为像素。
ImgHeight	Integer	是	原图高度，单位为像素。
FaceInfos	Map	是	人脸检测结果。具体结构描述见 FaceInfos 。

表 2-4: FaceInfos

名称	类型	是否必需	描述
FaceAttributesDetectInfo	Map	是	人脸属性检测结果。具体结构描述见 FaceAttributesDetectInfo 。

表 2-5: FaceAttributesDetectInfo

名称	类型	是否必需	描述
FaceAttributes	Map	是	人脸属性。具体结构描述见 FaceAttributes 。
FaceRect	Map	是	人脸在原图中的位置。具体结构描述见 FaceRect 。

表 2-6: FaceAttributes

名称	类型	是否必需	描述
Facetype	String	否	是否人脸，取值： - None：不是人脸 - Face：人脸 - Profile：侧脸（左右转头超过30°）  说明： 若没有检测到脸，接口返回的faceInfos即为空；这里的None代表检测到脸，但认为是卡通或宠物等。
Headpose	Map	否	人脸姿态。具体结构描述见 Headpose 。
Glasses	String	否	是否戴眼镜，取值： - None：未戴眼镜 - Wear：戴普通眼镜 - Sunglass：戴墨镜
Blur	Float	否	人脸模糊度，分值越大越模糊。通常 ≥ 2.0 已较模糊，建议根据实际业务数据测试调整。
Smiling	Map	否	是否笑脸。具体结构描述见 Smiling 。
Gender	Map	否	性别。具体结构描述见 Gender 。
Age	Integer	否	年龄。

表 2-7: Headpose

名称	类型	是否必需	描述
PitchAngle	Float	是	抬头角度，单位为弧度。
RollAngle	Float	是	左右歪头角度，单位为弧度。
YawAngle	Float	是	左右转头角度，单位为弧度。

表 2-8: Gender

名称	类型	是否必需	描述
Value	String	是	性别，取值：

名称	类型	是否必需	描述
			- Female：女性 - Male：男性
Score	Float	是	性别置信度，分值越大越可信。通常以 Value 值为准即可。

表 2-9: Smiling

名称	类型	是否必需	描述
Value	Float	是	笑脸分值。
Threshold	Float	是	笑脸阈值。

表 2-10: FaceRect

名称	类型	是否必需	描述
Left	Integer	是	人脸矩形左上角距原图左边的距离，单位为像素。
Top	Integer	是	人脸矩形左上角距原图上边的距离，单位为像素。
Height	Integer	是	人脸矩形的高度。
Width	Integer	是	人脸矩形的宽度。

错误码

错误代码	英文描述	中文描述	HTTP 状态码
Error.InternalError	The request processing has failed due to some unknown error.	服务内部出现未知错误，请联系技术支持排查。	500
InvalidParam. ParamMissing	Please check if there is param empty in your input.	参数错误，请检查参数是否为空。	400
Error.NoPermissionAccess	No permission to access our service.	没有权限访问服务，请确认是否服务已开通。	403

示例

关于使用SDK开发包的示例，请参考以下具体SDK 开发包使用文档中的认证方案说明：

以下是拼接**HTTPS**请求的相关示例：

请求示例

```
https://cloudauth.aliyuncs.com/?Action=DetectFaceAttributes
&RegionId=cn-hangzhou
&<[公共请求参数]>

请求body
MaterialValue=http%3A%2F%2Fjiangsu.china.com.cn%2Fuploadfile%2F2015%
2F0114%2F1421221304095989.jpg
&RetAttributes=gender%2cheadpose%2cblur%2cglasses%2cage%2cfacetype
&MaxFaceNum=1
&DontSaveDB=true
&ClientTag=
&MaxNumPhotosPerCategory=1
```

返回示例

- JSON格式

```
{
  "Data": {
    "ImgWidth": 1080,
    "ImgHeight": 1920,
    "FaceInfos": {
      "FaceAttributesDetectInfo": [
        {
          "FaceAttributes": {
            "Facetype": "Face",
            "Headpose": {
              "PitchAngle": -1.5683923959732056,
              "RollAngle": 7.163370132446289,
              "YawAngle": -6.925303936004639
            },
            "Glasses": "None",
            "Blur": 0.1419367790222168,
            "Smiling": {},
            "Gender": {
              "Score": 0.7767378687858582,
              "Value": "Female"
            },
            "Age": 2
          },
          "FaceRect": {
            "Left": 354,
            "Top": 453,
            "Height": 473,
            "Width": 473
          }
        }
      ]
    }
  },
  "Code": "1",
  "Success": true
}
```



```
}
```

2.5 使用SDK开发包

2.5.1 Java SDK

获取地址

- `aliyun-java-sdk-core` : [mvnrepository](#) / [maven.org](#) / [GitHub](#)
- `aliyun-java-sdk-cloudauth` : [mvnrepository](#) / [maven.org](#) / [GitHub](#)



说明：

两个SDK都必须引入，其中`aliyun-java-sdk-core`为阿里云的核心SDK，`aliyun-java-sdk-cloudauth`为实人认证的SDK。

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中额外加上以下内容：`profile.getHttpClientConfig().setIgnoreSSLCerts(true);`

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>4.1.2</version>
</dependency>
<dependency>
  <groupId>com.aliyun</groupId>
  <artifactId>aliyun-java-sdk-cloudauth</artifactId>
  <version>1.1.5</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认证方案示例

```
//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);
String biz = "YourRPBasicBiz"; //您在控制台上创建的、采用RPBasic认证方案的认
证场景标识，创建方法：https://help.aliyun.com/document\_detail/59975.html
String ticketId = UUID.randomUUID().toString(); //认证ID，由使用方指
定，发起不同的认证任务需要更换不同的认证ID
String token = null; //认证token，表达一次认证会话
int statusCode = -1; //-1 未认证，0 认证中，1 认证通过，2 认证不通过
//1. 服务端发起认证请求，获取到token
//GetVerifyToken接口文档：https://help.aliyun.com/document\_detail/57050.html
GetVerifyTokenRequest getVerifyTokenRequest = new GetVerifyTokenRequest();
getVerifyTokenRequest.setBiz(biz);
getVerifyTokenRequest.setTicketId(ticketId);
try {
    GetVerifyTokenResponse response = client.getAcsResponse(getVerifyTokenRequest);
    token = response.getData().getVerifyToken().getToken(); //token默认
    30分钟时效，每次发起认证时都必须实时获取
} catch (ServerException e) {
    e.printStackTrace();
} catch (ClientException e) {
    e.printStackTrace();
}
//2. 服务端将token传递给无线客户端
//3. 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. 服务端查询认证状态(客户端回调中也会携带认证状态，但建议以服务端调接口确认的为准)
//GetStatus接口文档：https://help.aliyun.com/document\_detail/57049.html
GetStatusRequest getStatusRequest = new GetStatusRequest();
getStatusRequest.setBiz(biz);
getStatusRequest.setTicketId(ticketId);
try {
```

```

        GetStatusResponse response = client.getAcsResponse(getStatusR
equest);
        statusCode = response.getData().getStatusCode();
    } catch (ServerException e) {
        e.printStackTrace();
    } catch (ClientException e) {
        e.printStackTrace();
    }
}
//7. 服务端获取认证资料
//GetMaterials接口文档: https://help.aliyun.com/document_detail/57641.
html
GetMaterialsRequest getMaterialsRequest = new GetMaterialsRequest();
getMaterialsRequest.setBiz(biz);
getMaterialsRequest.setTicketId(ticketId);
if( 1 == statusCode || 2 == statusCode ) { //认证通过or认证不通过
    try {
        GetMaterialsResponse response = client.getAcsResponse(
getMaterialsRequest);
        //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

```

RPMannual认证方案示例

代码示例同[RPBasic](#)。其中**biz**须传入您在控制台上创建的、采用RPMannual认证方案的场景标识。

RPBioID认证方案示例

```

//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);
String biz = "YourRPBioIDBiz"; //您在控制台上创建的、采用RPBioID认证方案的认
证场景标识, 创建方法: https://help.aliyun.com/document_detail/59975.html
String ticketId = UUID.randomUUID().toString(); //认证ID, 由使用方指
定, 发起不同的认证任务需要更换不同的认证ID
String token = null; //认证token, 表达一次认证会话
int statusCode = -1; //-1 未认证, 0 认证中, 1 认证通过, 2 认证不通过
//1. 服务端发起认证请求, 获取到token
//GetVerifyToken接口文档: https://help.aliyun.com/document_detail/57050.
html
GetVerifyTokenRequest getVerifyTokenRequest = new GetVerifyTokenReques
t();
getVerifyTokenRequest.setBiz(biz);
getVerifyTokenRequest.setTicketId(ticketId);
getVerifyTokenRequest.setMethod(MethodType.POST);
//通过binding参数传入业务已经采集的认证资料, 其中姓名、身份证号为必要字段
//若需要binding图片资料, 请控制单张图片大小在 2M 内, 避免拉取超时

```

```

getVerifyTokenRequest.setBinding("{\"Name\": \"张三\", \"IdentificationNumber\": \"330110201711110101\"}");
try {
    GetVerifyTokenResponse response = client.getAcsResponse(getVerifyTokenRequest);
    token = response.getData().getVerifyToken().getToken(); //token默认
    30分钟时效，每次发起认证时都必须实时获取
} catch (ServerException e) {
    e.printStackTrace();
} catch (ClientException e) {
    e.printStackTrace();
}
//2. 服务端将token传递给无线客户端
//3. 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. 服务端查询认证状态(客户端回调中也会携带认证状态，但建议以服务端调接口确认的为准)
//GetStatus接口文档：https://help.aliyun.com/document_detail/57049.html
GetStatusRequest getStatusRequest = new GetStatusRequest();
getStatusRequest.setBiz(biz);
getStatusRequest.setTicketId(ticketId);
try {
    GetStatusResponse response = client.getAcsResponse(getStatusRequest);
    statusCode = response.getData().getStatusCode();
} catch (ServerException e) {
    e.printStackTrace();
} catch (ClientException e) {
    e.printStackTrace();
}
//7. 服务端获取认证资料
//GetMaterials接口文档：https://help.aliyun.com/document_detail/57641.html
GetMaterialsRequest getMaterialsRequest = new GetMaterialsRequest();
getMaterialsRequest.setBiz(biz);
getMaterialsRequest.setTicketId(ticketId);
if( 1 == statusCode || 2 == statusCode ) { //认证通过or认证不通过
    try {
        GetMaterialsResponse response = client.getAcsResponse(getMaterialsRequest);
        //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

```

RPBioOnlyPro认证方案示例

代码示例同[RPBioID](#)。其中**biz**须传入您在控制台上创建的、采用RPBioOnlyPro认证方案的场景标识。

RPBioOnly认证方案示例

代码示例同[RPBioID](#)。其中**biz**须传入您在控制台上创建的、采用RPBioOnly认证方案的场景标识。

FVBioOnly认证方案示例

```
//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);
String biz = "YourFVBioOnlyBiz"; //您在控制台上创建的、采用FVBioOnly认证方案的认证场景标识，创建方法：https://help.aliyun.com/document\_detail/59975.html
String ticketId = UUID.randomUUID().toString(); //认证ID，由使用方指定，发起不同的认证任务需要更换不同的认证ID
String token = null; //认证token，表达一次认证会话
int statusCode = -1; //-1 未认证，0 认证中，1 认证通过，2 认证不通过
//1. 服务端发起认证请求，获取到token
//GetVerifyToken接口文档：https://help.aliyun.com/document\_detail/57050.html
GetVerifyTokenRequest getVerifyTokenRequest = new GetVerifyTokenRequest();
getVerifyTokenRequest.setBiz(biz);
getVerifyTokenRequest.setTicketId(ticketId);
getVerifyTokenRequest.setMethod(MethodType.POST);
//通过binding参数传入业务已经采集的认证资料，其中人像留底照片为必要字段
//请控制单张图片大小在 2M 内，避免拉取超时
getVerifyTokenRequest.setBinding("{\"FaceRetainedPic\": \"http://...\"}");
try {
    GetVerifyTokenResponse response = client.getAcsResponse(getVerifyTokenRequest);
    token = response.getData().getVerifyToken().getToken(); //token默认30分钟时效，每次发起认证时都必须实时获取
} catch (ServerException e) {
    e.printStackTrace();
} catch (ClientException e) {
    e.printStackTrace();
}
//2. 服务端将token传递给无线客户端
//3. 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. 服务端查询认证状态(客户端回调中也会携带认证状态，但建议以服务端调接口确认的为准)
//GetStatus接口文档：https://help.aliyun.com/document\_detail/57049.html
GetStatusRequest getStatusRequest = new GetStatusRequest();
getStatusRequest.setBiz(biz);
getStatusRequest.setTicketId(ticketId);
try {
```

```

        GetStatusResponse response = client.getAcsResponse(getStatusR
equest);
        statusCode = response.getData().getStatusCode();
    } catch (ServerException e) {
        e.printStackTrace();
    } catch (ClientException e) {
        e.printStackTrace();
    }
}
//7. 服务端获取认证资料
//GetMaterials接口文档: https://help.aliyun.com/document_detail/57641.
html
GetMaterialsRequest getMaterialsRequest = new GetMaterialsRequest();
getMaterialsRequest.setBiz(biz);
getMaterialsRequest.setTicketId(ticketId);
if( 1 == statusCode || 2 == statusCode ) { //认证通过or认证不通过
    try {
        GetMaterialsResponse response = client.getAcsResponse(
getMaterialsRequest);
        //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

```

RPH5BioOnly认证方案示例

```

//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);
String biz = "YourRPH5BioOnlyBiz"; //您在控制台上创建的、采用RPH5BioOnly
认证方案的认证场景标识, 创建方法: https://help.aliyun.com/document_detail/
59975.html
String ticketId = UUID.randomUUID().toString(); //认证ID, 由使用方指
定, 发起不同的认证任务需要更换不同的认证ID
String token = null; //认证token, 表达一次认证会话
int statusCode = -1; //-1 未认证, 0 认证中, 1 认证通过, 2 认证不通过
//1. 服务端发起认证请求, 获取到token
//GetVerifyToken接口文档: https://help.aliyun.com/document_detail/57050.
html
GetVerifyTokenRequest getVerifyTokenRequest = new GetVerifyTokenReques
t();
getVerifyTokenRequest.setBiz(biz);
getVerifyTokenRequest.setTicketId(ticketId);
getVerifyTokenRequest.setMethod(MethodType.POST);
//通过binding参数传入业务已经采集的认证资料, 其中姓名、身份证号为必要字段
//若需要binding图片资料, 请控制单张图片大小在 2M 内, 避免拉取超时
getVerifyTokenRequest.setBinding("{\"Name\": \"张三\", \"Identifica
tionNumber\": \"330110201711110101\"}");
try {
    GetVerifyTokenResponse response = client.getAcsResponse(getVerifyT
okenRequest);
}

```

```

        token = response.getData().getVerifyToken().getToken(); //token默认
        30分钟时效，每次发起认证时都必须实时获取
    } catch (ServerException e) {
        e.printStackTrace();
    } catch (ClientException e) {
        e.printStackTrace();
    }
}
//2. 服务端将认证URL(带token)传递给H5前端
//3. H5前端跳转认证URL
//4. 用户按照认证H5流程页面的指引，提交认证资料
//5. 认证流程结束跳转指定的重定向URL(指定方法参看：https://help.aliyun.com/document\_detail/58644.html?#H5Server)
//6. 服务端查询认证状态(建议以服务端调接口确认的为准)
//GetStatus接口文档：https://help.aliyun.com/document\_detail/57049.html
GetStatusRequest getStatusRequest = new GetStatusRequest();
getStatusRequest.setBiz(biz);
getStatusRequest.setTicketId(ticketId);
try {
    GetStatusResponse response = client.getAcsResponse(getStatusRequest);
    statusCode = response.getData().getStatusCode();
} catch (ServerException e) {
    e.printStackTrace();
} catch (ClientException e) {
    e.printStackTrace();
}
}
//7. 服务端获取认证资料
//GetMaterials接口文档：https://help.aliyun.com/document\_detail/57641.html
GetMaterialsRequest getMaterialsRequest = new GetMaterialsRequest();
getMaterialsRequest.setBiz(biz);
getMaterialsRequest.setTicketId(ticketId);
if( 1 == statusCode || 2 == statusCode ) { //认证通过or认证不通过
    try {
        GetMaterialsResponse response = client.getAcsResponse(getMaterialsRequest);
        //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

```

RPMIn认证方案示例

```

//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);
String biz = "YourRPMInBiz"; //您在控制台上创建的、采用RPMIn认证方案的认证场景标识，创建方法：https://help.aliyun.com/document\_detail/59975.html
String ticketId = UUID.randomUUID().toString(); //认证ID，由使用方指定，发起不同的认证任务需要更换不同的认证ID

```

```
String token = null; //认证token, 表达一次认证会话
//1. 发起认证请求, 获取到token
//GetVerifyToken接口文档: https://help.aliyun.com/document_detail/57050.html
GetVerifyTokenRequest getVerifyTokenRequest = new GetVerifyTokenRequest();
getVerifyTokenRequest.setBiz(biz); //传入采用RPMIn认证方案的认证场景标识(biz)
getVerifyTokenRequest.setTicketId(ticketId);
try {
    GetVerifyTokenResponse response = client.getAcsResponse(getVerifyTokenRequest);
    token = response.getData().getVerifyToken().getToken(); //token默认30分钟时效, 每次发起认证时都必须实时获取
} catch (Exception e) {
    e.printStackTrace();
}
//2. 用token提交认证材料
//SubmitMaterials接口文档: https://help.aliyun.com/document_detail/58176.html
SubmitMaterialsRequest submitRequest = new SubmitMaterialsRequest();
submitRequest.setVerifyToken(token);
submitRequest.setMethod(MethodType.POST);
//创建要提交的认证材料列表, 请根据 认证方案 中的说明传入相应字段
List<SubmitMaterialsRequest.Material> verifyMaterials = new ArrayList<SubmitMaterialsRequest.Material>();
SubmitMaterialsRequest.Material identificationNumber = new SubmitMaterialsRequest.Material();
identificationNumber.setMaterialType("IdentificationNumber");
identificationNumber.setValue("330110201711110101");
verifyMaterials.add(identificationNumber);
SubmitMaterialsRequest.Material name = new SubmitMaterialsRequest.Material();
name.setMaterialType("Name");
name.setValue("张三");
verifyMaterials.add(name);
//传入图片资料, 请控制单张图片大小在 2M 内, 避免拉取超时
SubmitMaterialsRequest.Material facePic = new SubmitMaterialsRequest.Material();
facePic.setMaterialType("FacePic");
facePic.setValue("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"data:image/png;base64,"), 并注意控制接口请求的Body在8M以内
verifyMaterials.add(facePic);
SubmitMaterialsRequest.Material idCardFrontPic = new SubmitMaterialsRequest.Material();
idCardFrontPic.setMaterialType("IdCardFrontPic");
idCardFrontPic.setValue("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
verifyMaterials.add(idCardFrontPic);
SubmitMaterialsRequest.Material idCardBackPic = new SubmitMaterialsRequest.Material();
idCardBackPic.setMaterialType("IdCardBackPic");
idCardBackPic.setValue("oss://verify-img:715559d76a407740SS.JPG"); //oss方式上传图片, 此oss文件地址须可公开访问
verifyMaterials.add(idCardBackPic);
submitRequest.setMaterials(verifyMaterials);
try {
```



```

    SubmitMaterialsResponse response = client.getAcsResponse(
submitRequest);
    //由于审核需要时间，SubmitMaterials接口并不保证同步返回认证结果，可能会返回
认证中状态，此时需要使用GetStatus接口轮询认证结果。
    //GetStatus接口文档：https://help.aliyun.com/document_detail/57049.
html
    //GetStatusRequest getStatusRequest = new GetStatusRequest();
    //getStatusRequest.setBiz(biz);
    //getStatusRequest.setTicketId(ticketId);
    //GetStatusResponse response = client.getAcsResponse(getStatusR
equest);
    //int statusCode = response.getData().getStatusCode();
    //后续业务处理
} catch (Exception e) {
    e.printStackTrace();
}
//常见问题：https://help.aliyun.com/document_detail/57640.html

```

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
串去掉头部描述(如"data:image/png;base64,")，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

```

2.5.2 PHP SDK

获取地址

[GitHub](#)，引入`aliyun-php-sdk-core`和`aliyun-php-sdk-cloudauth`。

**说明：**

两个SDK 都必须引入。其中`aliyun-php-sdk-core`为阿里云的核心SDK，`aliyun-php-sdk-cloudauth`为实人认证的SDK。

使用说明

1. 从GitHub中下载PHP SDK的源代码。
2. 在代码中根据实际文件路径添加PHP SDK引用，参看下方代码示例。

RPBasic认证方案示例

```
<?php
include_once './aliyun-php-sdk-core/Config.php';
include_once 'Guid.php'; //参见https://help.aliyun.com/document_detail/64081.html#guid
use Cloudauth\Request\V20180504 as cloudauth; //请以实际目录为准
// 创建DefaultAcsClient实例并初始化
$iClientProfile = DefaultProfile::getProfile(
    "cn-hangzhou",           //默认
    "YourAccessKeyId",       //您的Access Key ID
    "YourAccessKeySecret");  //您的Access Key Secret
$iClientProfile::addEndpoint("cn-hangzhou", "cn-hangzhou", "Cloudauth", "cloudauth.aliyuncs.com");
$client = new DefaultAcsClient($iClientProfile);
$biz = "YourRPBasicBiz"; //您在控制台上创建的、采用RPBasic认证方案的认证场景标识，创建方法：https://help.aliyun.com/document_detail/59975.html
$ticketId = guid(); //认证ID，由使用方指定，发起不同的认证任务需要更换不同的认证ID
$token = null; //认证token，表达一次认证会话
$statuscode = -1; //-1 未认证，0 认证中，1 认证通过，2 认证不通过
//1. 服务端发起认证请求，获取到token
//GetVerifyToken接口文档：https://help.aliyun.com/document_detail/57050.html
$getVerifyTokenRequest = new cloudauth\GetVerifyTokenRequest();
$getVerifyTokenRequest->setBiz($biz);
$getVerifyTokenRequest->setTicketId($ticketId);
try {
    $response = $client->getAcsResponse($getVerifyTokenRequest);
    $token = $response->Data->VerifyToken->Token; //token默认30分钟时效，每次发起认证时都必须实时获取
} catch (Exception $e) {
    print $e->getTrace();
}
//2. 服务端将token传递给无线客户端
//3. 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. 服务端查询认证状态(客户端回调中也会携带认证状态，但建议以服务端调接口确认的为准)
//GetStatus接口文档：https://help.aliyun.com/document_detail/57049.html
$getStatusRequest = new cloudauth\GetStatusRequest();
```

```

$getStatusRequest->setBiz($biz);
$getStatusRequest->setTicketId($ticketId);
try {
    $response = $client->getAcsResponse($getStatusRequest);
    $statusCode = $response->Data->StatusCode;
} catch (ServerException $e) {
    print $e->getMessage();
} catch (ClientException $e) {
    print $e->getMessage();
}
}
//7. 服务端获取认证资料
//GetMaterials接口文档 : https://help.aliyun.com/document_detail/57641.html
$getMaterialsRequest = new cloudauth\GetMaterialsRequest();
$getMaterialsRequest->setBiz($biz);
$getMaterialsRequest->setTicketId($ticketId);
if (1 == $statusCode or 2 == $statusCode) { //认证通过or认证不通过
    try {
        $response = $client->getAcsResponse($getMaterialsRequest);
    } catch (ServerException $e) {
        print $e->getMessage();
    } catch (ClientException $e) {
        print $e->getMessage();
    }
}
}
//常见问题 : https://help.aliyun.com/document_detail/57640.html

```

RPMannual认证方案示例

代码示例同[RPBasic](#)。其中**biz**须传入您在控制台上创建的、采用RPMannual认证方案的场景标识。

RPBioID认证方案示例

```

<?php
include_once './aliyun-php-sdk-core/Config.php';
include_once 'Guid.php'; //参见https://help.aliyun.com/document_detail/64081.html#guid
use Cloudauth\Request\V20180504 as cloudauth; //请以实际目录为准
// 创建DefaultAcsClient实例并初始化
$iClientProfile = DefaultProfile::getProfile(
    "cn-hangzhou",           //默认
    "YourAccessKeyId",       //您的Access Key ID
    "YourAccessKeySecret");  //您的Access Key Secret
$iClientProfile::addEndpoint("cn-hangzhou", "cn-hangzhou", "Cloudauth", "cloudauth.aliyuncs.com");
$client = new DefaultAcsClient($iClientProfile);
$biz = "YourRPBioIDBiz"; //您在控制台上创建的、采用RPBioID认证方案的认证场景标识, 创建方法 : https://help.aliyun.com/document_detail/59975.html
$ticketId = guid(); //认证ID, 由使用方指定, 发起不同的认证任务需要更换不同的认证ID
$token = null; //认证token, 表达一次认证会话
$statusCode = -1; //-1 未认证, 0 认证中, 1 认证通过, 2 认证不通过
//1. 服务端发起认证请求, 获取到token
//GetVerifyToken接口文档 : https://help.aliyun.com/document_detail/57050.html
$getVerifyTokenRequest = new cloudauth\GetVerifyTokenRequest();

```

```

$getVerifyTokenRequest->setBiz($biz);
$getVerifyTokenRequest->setTicketId($ticketId);
//若需要binding图片(如身份证正反面等), 且使用base64上传, 需要设置请求方法为POST
$getVerifyTokenRequest->setMethod("POST");
//通过binding参数传入业务已经采集的认证资料, 其中姓名、身份证号为必要字段
//若需要binding图片资料, 请控制单张图片大小在 2M 内, 避免拉取超时
$getVerifyTokenRequest->setBinding("{\"Name\": \"张三\", \"IdentificationNumber\": \"330110201711110101\"}");
try {
    $response = $client->getAcsResponse($getVerifyTokenRequest);
    $token = $response->Data->VerifyToken->Token; //token默认30分钟时效, 每次发起认证时都必须实时获取
} catch (Exception $e) {
    print $e->getTrace();
}
//2. 服务端将token传递给无线客户端
//3. 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. 服务端查询认证状态(客户端回调中也会携带认证状态, 但建议以服务端调接口确认的为准)
//GetStatus接口文档: https://help.aliyun.com/document_detail/57049.html
$statusRequest = new cloudauth\GetStatusRequest();
$statusRequest->setBiz($biz);
$statusRequest->setTicketId($ticketId);
try {
    $response = $client->getAcsResponse($getStatusRequest);
    $statusCode = $response->Data->StatusCode;
} catch (ServerException $e) {
    print $e->getMessage();
} catch (ClientException $e) {
    print $e->getMessage();
}
//7. 服务端获取认证资料
//GetMaterials接口文档: https://help.aliyun.com/document_detail/57641.html
$materialsRequest = new cloudauth\GetMaterialsRequest();
$materialsRequest->setBiz($biz);
$materialsRequest->setTicketId($ticketId);
if (1 == $statusCode or 2 == $statusCode) { //认证通过or认证不通过
    try {
        $response = $client->getAcsResponse($getMaterialsRequest);
    } catch (ServerException $e) {
        print $e->getMessage();
    } catch (ClientException $e) {
        print $e->getMessage();
    }
}
//常见问题: https://help.aliyun.com/document_detail/57640.html

```

RPBioOnlyPro认证方案示例

代码示例同 [RPBioID](#)。其中**biz**须传入您在控制台上创建的、采用RPBioOnlyPro认证方案的场景标识。

RPBioOnly认证方案示例

代码示例同[RPBioID](#)。其中**biz**须传入您在控制台上创建的、采用RPBioOnly认证方案的场景标识。

FVBioOnly认证方案示例

```
<?php
include_once './aliyun-php-sdk-core/Config.php';
include_once 'Guid.php'; //参见https://help.aliyun.com/document_detail/64081.html#guid
use Cloudauth\Request\V20180504 as cloudauth; //请以实际目录为准
// 创建DefaultAcsClient实例并初始化
$iClientProfile = DefaultProfile::getProfile(
    "cn-hangzhou",           //默认
    "YourAccessKeyID",       //您的Access Key ID
    "YourAccessKeySecret");  //您的Access Key Secret
$iClientProfile->addEndpoint("cn-hangzhou", "cn-hangzhou", "Cloudauth", "cloudauth.aliyuncs.com");
$client = new DefaultAcsClient($iClientProfile);
$biz = "YourFVBioOnlyBiz"; //您在控制台上创建的、采用FVBioOnly认证方案的认证场景标识，创建方法：https://help.aliyun.com/document_detail/59975.html
$ticketId = guid(); //认证ID，由使用方指定，发起不同的认证任务需要更换不同的认证ID
$token = null; //认证token，表达一次认证会话
$statuscode = -1; //-1 未认证，0 认证中，1 认证通过，2 认证不通过
//1. 服务端发起认证请求，获取到token
//GetVerifyToken接口文档：https://help.aliyun.com/document_detail/57050.html
$getVerifyTokenRequest = new cloudauth\GetVerifyTokenRequest();
$getVerifyTokenRequest->setBiz($biz);
$getVerifyTokenRequest->setTicketId($ticketId);
//若需要使用base64上传图片，必须设置请求方法为POST
$getVerifyTokenRequest->setMethod("POST");
//通过binding参数传入业务已经采集的认证资料，其中人像留底照片为必要字段
//请控制单张图片大小在 2M 内，避免拉取超时
$getVerifyTokenRequest->setBinding("{\"FaceRetainedPic\": \"http://...\"}");
try {
    $response = $client->getAcsResponse($getVerifyTokenRequest);
    $token = $response->Data->VerifyToken->Token; //token默认30分钟时效，每次发起认证时都必须实时获取
} catch (Exception $e) {
    print $e->getTrace();
}
//2. 服务端将token传递给无线客户端
//3. 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. 服务端查询认证状态(客户端回调中也会携带认证状态，但建议以服务端调接口确认的为准)
//GetStatus接口文档：https://help.aliyun.com/document_detail/57049.html
$statusRequest = new cloudauth\GetStatusRequest();
$statusRequest->setBiz($biz);
```

```

$statusRequest->setTicketId($ticketId);
try {
    $response = $client->getAcsResponse($getStatusRequest);
    $statusCode = $response->Data->StatusCode;
} catch (ServerException $e) {
    print $e->getMessage();
} catch (ClientException $e) {
    print $e->getMessage();
}
}
//7. 服务端获取认证资料
//GetMaterials接口文档: https://help.aliyun.com/document_detail/57641.html
$getStatusRequest = new cloudauth\GetMaterialsRequest();
$getStatusRequest->setBiz($biz);
$getStatusRequest->setTicketId($ticketId);
if (1 == $statusCode or 2 == $statusCode) { //认证通过or认证不通过
    try {
        $response = $client->getAcsResponse($getStatusRequest);
    } catch (ServerException $e) {
        print $e->getMessage();
    } catch (ClientException $e) {
        print $e->getMessage();
    }
}
}
//常见问题: https://help.aliyun.com/document_detail/57640.html

```

RPH5BioOnly认证方案示例

```

<?php
include_once './aliyun-php-sdk-core/Config.php';
include_once 'Guid.php'; //参见https://help.aliyun.com/document_detail/64081.html#guid
use Cloudauth\Request\V20180504 as cloudauth; //请以实际目录为准
// 创建DefaultAcsClient实例并初始化
$iClientProfile = DefaultProfile::getProfile(
    "cn-hangzhou", //默认
    "YourAccessKeyID", //您的Access Key ID
    "YourAccessKeySecret"); //您的Access Key Secret
$iClientProfile::addEndpoint("cn-hangzhou", "cn-hangzhou", "Cloudauth", "cloudauth.aliyuncs.com");
$client = new DefaultAcsClient($iClientProfile);
$biz = "YourRPH5BioOnlyBiz"; //您在控制台上创建的、采用RPH5BioOnly认证方案的认证场景标识, 创建方法: https://help.aliyun.com/document_detail/59975.html
$ticketId = guid(); //认证ID, 由使用方指定, 发起不同的认证任务需要更换不同的认证ID
$token = null; //认证token, 表达一次认证会话
$statusCode = -1; //-1 未认证, 0 认证中, 1 认证通过, 2 认证不通过
//1. 服务端发起认证请求, 获取到token
//GetVerifyToken接口文档: https://help.aliyun.com/document_detail/57050.html
$verifyTokenRequest = new cloudauth\GetVerifyTokenRequest();
$verifyTokenRequest->setBiz($biz);
$verifyTokenRequest->setTicketId($ticketId);
//若需要binding图片(如身份证正面反面等), 且使用base64上传, 需要设置请求方法为POST
//$verifyTokenRequest->setMethod("POST");
//通过binding参数传入业务已经采集的认证资料, 其中姓名、身份证号为必要字段
//若需要binding图片资料, 请控制单张图片大小在 2M 内, 避免拉取超时

```

```

$getVerifyTokenRequest->setBinding("{\"Name\": \"张三\", \"IdentificationNumber\": \"330110201711110101\"}");
try {
    $response = $client->getAcsResponse($getVerifyTokenRequest);
    $token = $response->Data->VerifyToken->Token; //token默认30分钟时效, 每次发起认证时都必须实时获取
} catch (Exception $e) {
    print $e->getTrace();
}
//2. 服务端将认证URL(带token)传递给H5前端
//3. H5前端跳转认证URL
//4. 用户按照认证H5流程页面的指引, 提交认证资料
//5. 认证流程结束跳转指定的重定向URL(指定方法参看: https://help.aliyun.com/document_detail/58644.html?#H5Server)
//6. 服务端查询认证状态(建议以服务端调接口确认的为准)
//GetStatus接口文档: https://help.aliyun.com/document_detail/57049.html
$getStatusRequest = new cloudauth\GetStatusRequest();
$getStatusRequest->setBiz($biz);
$getStatusRequest->setTicketId($ticketId);
try {
    $response = $client->getAcsResponse($getStatusRequest);
    $statusCode = $response->Data->StatusCode;
} catch (ServerException $e) {
    print $e->getMessage();
} catch (ClientException $e) {
    print $e->getMessage();
}
//7. 服务端获取认证资料
//GetMaterials接口文档: https://help.aliyun.com/document_detail/57641.html
$getMaterialsRequest = new cloudauth\GetMaterialsRequest();
$getMaterialsRequest->setBiz($biz);
$getMaterialsRequest->setTicketId($ticketId);
if (1 == $statusCode or 2 == $statusCode) { //认证通过or认证不通过
    try {
        $response = $client->getAcsResponse($getMaterialsRequest);
    } catch (ServerException $e) {
        print $e->getMessage();
    } catch (ClientException $e) {
        print $e->getMessage();
    }
}
//常见问题: https://help.aliyun.com/document_detail/57640.html

```

RPMin认证方案示例

```

<?php
include_once './aliyun-php-sdk-core/Config.php';
include_once 'Guid.php'; //参见https://help.aliyun.com/document_detail/64081.html#guid
use Cloudauth\Request\V20180504 as cloudauth; //请以实际目录为准
//创建DefaultAcsClient实例并初始化
$iClientProfile = DefaultProfile::getProfile(
    "cn-hangzhou",           //默认
    "YourAccessKeyID",       //您的Access Key ID
    "YourAccessKeySecret");  //您的Access Key Secret

```



```

$iClientProfile::addEndpoint("cn-hangzhou", "cn-hangzhou", "Cloudauth", "cloudauth.aliyuncs.com");
$client = new DefaultAcsClient($iClientProfile);
$biz = "YourRPMInBiz"; //您在控制台上创建的、采用RPMIn认证方案的认证场景标识, 创建方法: https://help.aliyun.com/document_detail/59975.html
$ticketId = guid(); //认证ID, 由使用方指定, 发起不同的认证任务需要更换不同的认证ID
$token = null; //认证token, 表达一次认证会话
//1. 服务端发起认证请求, 获取到token
//GetVerifyToken接口文档: https://help.aliyun.com/document_detail/57050.html
$getVerifyTokenRequest = new cloudauth\GetVerifyTokenRequest();
$getVerifyTokenRequest->setBiz($biz);
$getVerifyTokenRequest->setTicketId($ticketId);
try {
    $response = $client->getAcsResponse($getVerifyTokenRequest);
    $token = $response->Data->VerifyToken->Token; //token默认30分钟时效, 每次发起认证时都必须实时获取
} catch (Exception $e) {
    print $e->getTrace();
}
//2. 用token提交认证材料
//SubmitMaterials接口文档: https://help.aliyun.com/document_detail/58176.html
$submitRequest = new cloudauth\SubmitMaterialsRequest();
$submitRequest->setVerifyToken($token);
//若使用base64上传图片, 需要设置请求方法为POST
$submitRequest->setMethod("POST");
$identificationNumber = array("MaterialType" => "IdentificationNumber", "Value" => "330110201711110101");
$name = array("MaterialType" => "Name", "Value" => "张三");
//传入图片资料, 请控制单张图片大小在 2M 内, 避免拉取超时
$facePic = array("MaterialType" => "FacePic", "Value" => "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"data:image/png;base64,"), 并注意控制接口请求的Body在8M以内
$idCardFrontPic = array("MaterialType" => "IdCardFrontPic", "Value" => "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
$idCardBackPic = array("MaterialType" => "IdCardBackPic", "Value" => "oss://verify-img:715559d76a407740SS.JPG"); //oss方式上传图片, 此oss文件地址须可公开访问
$verifyMaterials = array($identificationNumber, $name, $facePic, $idCardFrontPic, $idCardBackPic);
$submitRequest->setMaterials($verifyMaterials);
try {
    $submitMaterialsResponse = $client->getAcsResponse($submitRequest);
    //由于审核需要时间, SubmitMaterials接口并不保证同步返回认证结果, 可能会返回认证中状态, 此时需要使用GetStatus接口轮询认证结果。
    //GetStatus接口文档: https://help.aliyun.com/document_detail/57049.html
    $getStatusRequest = new cloudauth\GetStatusRequest();
    $getStatusRequest->setBiz($biz);
    $getStatusRequest->setTicketId($ticketId);
    $response = $client->getAcsResponse($getStatusRequest);
    $statusCode = $response->Data->StatusCode;

```



```

    //后续业务处理
} catch (ServerException $e) {
    print $e->getMessage();
} catch (ClientException $e) {
    print $e->getMessage();
}
//常见问题 : https://help.aliyun.com/document\_detail/57640.html

```

人脸比对验证示例

```

<?php
include_once './aliyun-php-sdk-core/Config.php';
use Cloudauth\Request\V20180504 as cloudauth; //请以实际目录为准
//创建DefaultAcsClient实例并初始化
$iClientProfile = DefaultProfile::getProfile(
    "cn-hangzhou",           //默认
    "YourAccessKeyId",       //您的Access Key ID
    "YourAccessKeySecret");  //您的Access Key Secret
$iClientProfile::addEndpoint("cn-hangzhou", "cn-hangzhou", "Cloudauth", "cloudauth.aliyuncs.com");
$client = new DefaultAcsClient($iClientProfile);
//创建API请求并设置参数
//CompareFaces接口文档 : https://help.aliyun.com/document\_detail/59317.html
$request = new cloudauth\CompareFacesRequest();
//若使用base64上传图片, 需要设置请求方法为POST
$request->setMethod("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"data:image/png;base64,"), 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 {
    $response = $client->getAcsResponse($request);
    // 后续业务处理
} catch (ServerException $e) {
    print $e->getMessage();
} catch (ClientException $e) {
    print $e->getMessage();
}
//常见问题 : https://help.aliyun.com/document\_detail/57640.html

```

附录

生成GUID

```

<?php
function guid(){
    if (function_exists('com_create_guid')){
        return com_create_guid();
    }
}

```

```
    }else{
        mt_srand((double)microtime()*10000); // optional for php 4.2.0
        and up.
        $charid = strtoupper(md5(uniqid(rand(), true)));
        $hyphen = chr(45); // "-"
        $uuid = chr(123) // "{"
            .substr($charid, 0, 8).$hyphen
            .substr($charid, 8, 4).$hyphen
            .substr($charid, 12, 4).$hyphen
            .substr($charid, 16, 4).$hyphen
            .substr($charid, 20, 12)
            .chr(125); // "}"
        return $uuid;
    }
}
```

2.5.3 Python SDK

获取地址

- [GitHub](#) , 引入 *aliyun-python-sdk-core* 和 *aliyun-python-sdk-cloudauth*。



说明：

两个SDK都必须引入。其中 *aliyun-python-sdk-core* 为阿里云的核心SDK , *aliyun-python-sdk-cloudauth* 为实人认证的SDK。

使用说明

参考GitHub中的说明，或者[阿里云SDK开发指南 - Python SDK](#)目录下的章节。

使用示例

完整示例正在准备中，请先参考[Java SDK示例](#)。

2.5.4 .NET SDK

获取地址

- [GitHub](#) , 引入 *aliyun-net-sdk-core* 和 *aliyun-net-sdk-cloudauth*。



说明：

两个SDK都必须引入，其中 *aliyun-net-sdk-core* 为阿里云的核心SDK , *aliyun-net-sdk-cloudauth* 为实人认证的SDK。

使用说明

参考GitHub中的说明，或者[阿里云SDK开发指南 - .NET SDK](#)目录下的章节。

使用示例

完整示例正在准备中，请先参考[Java SDK 示例](#)。

2.5.5 Node.js SDK

获取地址

- [GitHub](#)，基于核心SDK进行开发，使用RPC调用风格。

使用说明

参考GitHub中的说明，或者[阿里云 SDK 开发指南 - Node.js SDK](#)目录下的章节。

使用示例

完整示例正在准备中，请先参考[Java SDK 示例](#)。

2.5.6 Go SDK

获取地址

- [GitHub](#)，使用泛化调用接口（CommonAPI）。

使用说明

参考GitHub中的说明，或者[阿里云 SDK 开发指南 - Go SDK](#)目录下的章节。

使用示例

完整示例正在准备中，请先参考[Java SDK 示例](#)。

3 无线认证SDK

3.1 下载无线SDK

介绍了如何下载可以集成到您的应用中的无线认证SDK。

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，请参考[访问控制](#)。

2. 在左侧导航栏选择接入及设置。
3. 在实人认证页，选择要操作的认定场景，然后单击获取认证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，而应用自身的版本迭代无需重新生成SDK。

6. 应用上传完成后，单击下载认证SDK完成下载。
7. 参考以下文档完成无线认证SDK的集成：

- [Android应用集成SDK](#)
- [iOS应用集成SDK](#)

3.2 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。

Android SDK 与包名(package name)+签名(keystore)绑定，package name和keystore两者修改任一，都需要在[管理控制台](#)上重新下载 SDK，debug 和 release 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（只选 V2 签名，apk 在 Android 7.0 以下都无法安装）。

步骤一：在工程导入SDK

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：

- `aliyun-oss-sdk-android.jar`
- `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`
- `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 files('../libs/okhttp-3.2.0.jar')
compile files('../libs/okio-1.6.0.jar')
compile files('../libs/aliyun-oss-sdk-android-2.2.0.jar')
compile files('../libs/windvane-min-8.0.3.2.3.jar')
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

1. 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。
2. 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。
3. 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 和 x86 四种，请接入方按需在建build.gradle中增加abiFilters配置。例如，接入方仅需要支持其中 armeabi、arm64-v8a 和 x86 三种，则配置如下：

```
defaultConfig {  
    ndk {  
        abiFilters "armeabi", "arm64-v8a", "x86"  
    }  
}
```

步骤二：添加 ProGuard 配置

如果您的应用使用了 ProGuard 进行代码混淆，为了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.**{*;}

```

步骤三：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) {
            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_IN_AUDIT) { //认证中，通常
不会出现，只有在认证审核系统内部出现超时，未在限定时间内返回认证结果时出现。此时提示
用户系统处理中，稍后查看认证结果即可。
            }
            else if(audit == RPSDK.AUDIT.AUDIT_NOT) { //未认证，用户取消
            }
            else if(audit == RPSDK.AUDIT.AUDIT_EXCEPTION){ //系统异常
            }
        }
    });

```



说明：

- verifyToken参数由接入方的服务端调用实人认证服务的[GetVerifyToken](#)接口获得。
- 在RPSDK.start 接口调用的回调会返回用户认证完之后的各种状态。

常见问题

- 调起无线认证SDK，进入认证页面展示“UNKNOWN_ERROR”。

处理建议：查看logcat，若 ErrorCode = 1208 或 1215 或 1411，说明当前开发包与在管理控制台下载 SDK 时上传包的包名(package name)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。



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、aliyun-oss-sdk-android 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 aliyun-oss-sdk-android 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。

3.3 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。



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。

步骤一：在工程导入 SDK

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：

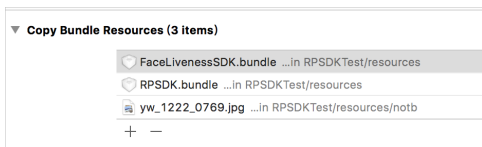
- *RPSDK.framework*
- *FaceLivenessOpen.framework*

- *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，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**。
- 将**Enable Bitcode**设为**No**。



步骤二：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:^(AUDIT auditState) {
    NSLog(@"verifyResult = %ld", (unsigned long)auditState);
    if(auditState == AUDIT_PASS) { //认证通过
    }
    else if(auditState == AUDIT_FAIL) { //认证不通过
    }
    else if(auditState == AUDIT_IN_AUDIT) { //认证中，通常不会出现，只有在认证审核系统内部出现超时，未在限定时间内返回认证结果时出现。此时提示用户系统处理中，稍后查看认证结果即可。
    }
    else if(auditState == AUDIT_NOT) { //未认证，用户取消
    }
    else if(auditState == AUDIT_EXCEPTION) { //系统异常
    }
}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[GetVerifyToken](#)接口获得。
- 认证的结果由回调的方式返回，并携带状态。

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 或 1215 或 1411，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。

- 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。

4 定向发布

4.1 提交认证

描述

接口名：**SubmitVerification**

请求方法：仅支持以HTTPS POST方法发送请求。

服务端提交认证材料到实名认证服务进行审核，结果同步返回。

关于上传图片地址的说明

在传入图片时，您需要上传其对应的HTTP/OSS地址或base64编码。

- HTTP地址：可访问的公网HTTP地址。例如，`http://image-demo.img-cn-hangzhou.aliyuncs.com/example.jpg`。
- OSS地址：使用STS功能上传图片到实名认证服务指定的OSS bucket后，所生成的OSS文件地址，格式为`oss://<STSToken.bucketName>:<path_to_file>`。[查看详情](#)。
- base64编码：通过base64编码的图片，格式为`base64://<图片 base64 字符串>`。



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。
- 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。
- 图片base64字符串需要去掉头部描述，如`data:image/png;base64,`。

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： <code>SubmitVerification</code> 。
RegionId	String	query	是	服务所在地域。取值： <code>cn-hangzhou</code> 。
Biz	String	body	是	使用实名认证服务的业务场景。请先参考 业务设置 在控制台完成创建。

名称	类型	参数位置	是否必需	描述
TicketId	String	body	是	标识一次认证任务的唯一ID。通常由业务使用方指定，方便关联业务场景的其他内容。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认证 ID。
Material.n. MaterialType	材料类型	body	是	提交审核的认证材料的类型。取值：  说明： 需要传入哪些字段取决于您接入时确定的认证方案，具体见认证方案中说明。 • Name：姓名。对应材料值为字符串格式，例如，张三。 • IdentificationNumber：身份证号。对应材料值为字符串格式，例如，630546198009090099。 • FacePic：人像照。对应材料值为字符串格式，具体请参考关于上传图片地址的说明。 • IdCardFrontPic：身份证人像面照片。对应材料值为字符串格式，具体请参考关于上传图片地址的说明。 • IdCardBackPic：身份证国徽面照片。对应材料值为字符串格式，具体请参考关于上传图片地址的说明。 • IDPic：二代身份证芯片中存储的头像照片。对应材料值为字符串格式，具体请参考关于上传图片地址的说明。

名称	类型	参数位置	是否必需	描述
				Mobile：手机号（该手机号用于辅助风险判断，而非与手机运营商实名信息进行核验。）对应取值为字符串格式，例如，13983822930。 n为整型可变参数，当需要传入多种类型的材料时，依次通过Material.1.MaterialTy、Material.2.MaterialTy、...、Material.10.MaterialTy传入类型值。n最大为10，且必须和Material.n.Value中的n匹配。
Material.n.Value	材料值	body	是	提交审核的认证材料的取值，与具体的材料类型对应，具体见Material.n.MaterialType描述。 n为整型可变参数，最大为10，且必须和Material.n.MaterialType中的n匹配。

返回参数

名称	类型	是否必需	描述
VerifyStatus	Map	是	提交认证资料后的认证状态。具体结构描述见 VerifyStatus 。

表 4-1: VerifyStatus

名称	类型	是否必需	描述
StatusCode	Integer	是	一个认证任务所处的认证状态。取值： -1：未认证。表示没有提交记录。 0：认证中。表示已提交认证，系统正在审核认证资料。 1：认证通过。表示最近一次提交的认证资料已通过审核，当前认证任务完结。 2：认证不通过。表示最近一次提交的认证资料未通过审核，当前认证任务还可以继续发起提交。

名称	类型	是否必需	描述
Similarity Score	Float	是	认证过程中所提交的人脸照片和身份证上的头像的相似程度分值。取值范围为[0,100]，分数越大相似度越高。  说明： 只有提交的认证资料中同时包含人脸照片和身份证人像面照片，该值才有意义。
AuditConclusions	String	是	认证状态为“认证不通过”时的原因描述。不通过原因包括但不限于以下几种情况： • 身份证照片模糊，光线问题造成字体无法识别。 • 身份证照片信息与认证提交信息不一致。 • 提交的照片非身份证照片。 建议您请按引导提交本人有效身份证照片。

错误码

错误代码	英文描述	中文描述	HTTP状态码
Error.InternalError	The request processing has failed due to some unknown error.	服务内部出现未知错误，请联系技术支持排查。	500
InvalidParam. ParamMissing	Please check if there is param empty in your input.	参数错误，请检查参数是否为空。	403
Error.NoPermissionAccess	No permission to access our service.	没有权限访问服务，请确认是否服务已开通。	403
InvalidParam. VerifyTokenExpired	The verify token was expired.	VerifyToken 过期。VerifyToken 是有时效性的，请确保在 token 失效前进行提交。	400
InvalidParam. MaterialsEmpty	The materials list is empty.	Materials 列表为空。	400
InvalidParam. MaterialsNotValid	The param materials is not valid.	请确保提交的材料类型正确性。	403

错误代码	英文描述	中文描述	HTTP状态码
Error.DuplicatedTicketId	A record associated with your ticketId has existed.	重复提交。一个 TicketId 参数已经提交，且认证状态为认证中或认证通过时，再次提交会出现此错误。	403
Error.SubmitMaterialsError	SubmitMaterials action failed due to internal error.	提交处理失败。通常是实人认证服务内部处理异常，请联系技术支持排查。	500
Error.MaterialsUploadError	The materials upload failed.	SubmitMaterials 在上传认证资料中出现错误，请检查图片地址格式正确，图片大小未超限，使用 base64 传图的已设置请求方法为POST。	500
Error.MaterialsInsufficient	The number of materials does not meet the requirements for verification.	提交认证的材料种类不满足要求，请检查上传的材料种类是否符合业务场景要求。	403

示例

请求示例

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

Material.1.MaterialType=FacePic
&Material.1.Value=http%3A%2F%2Fjiangsu.china.com.cn%2Fuploadfile%
2F2015%2F0114%2F1421221304095989.jpg
&Material.2.MaterialType=IDPic&SignatureNonce=c71f14e8-f2e7-45fd-9c5a-
721cf6b38c4a
&Material.2.Value=oss%3A%2F%2Fverify-img%3A715559d79b8d49c4aa
bfdad3f6a40774OSS.JPG
&Material.3.MaterialType=IdCardFrontPic&Signature=azM4ZgJ7auesjlWQuMBF
9X2Tnc8%3D
&Material.3.Value=oss%3A%2F%2Fverify-img%3A715559d79b8d49c4aa
bfdad3f6a40774OSS.JPG
&Material.4.MaterialType=IdCardBackPic
&Material.4.Value=oss%3A%2F%2Fverify-img%3A715559d79b8d49c4aa
bfdad3f6a40774OSS.JPG
```



```
&Material.5.MaterialType=IdentificationNumber
&Material.5.Value=630546198009090099
&Material.6.MaterialType=Name
&Material.6.Value=%E6%9D%8E%E4%B8%89
```

返回示例

- XML格式

```
<?xml version='1.0' encoding='UTF-8'?>
<SubmitVerificationResponse>
  <Data>
    <VerifyStatus>
      <StatusCode>1</StatusCode>
      <SimilarityScore>93.12</SimilarityScore>
      <AuditConclusions>不通过原因</AuditConclusions>
    </VerifyStatus>
  </Data>
  <Success>true</Success>
  <Code>1</Code>
</SubmitVerificationResponse>
```

- JSON格式

```
{
  "Code": "1",
  "Data": {
    "VerifyStatus": {
      "MaterialScore": 95.22905,
      "StatusCode": 1
      "AuditConclusions": "不通过原因"
    }
  },
  "Success": true
}
```

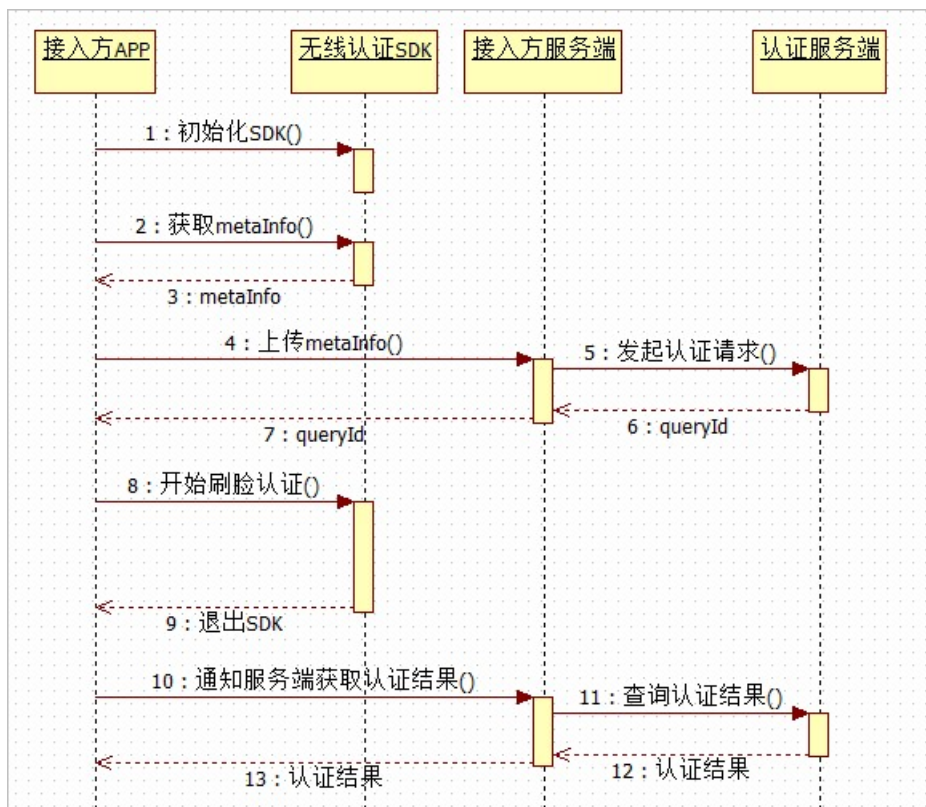
4.2 实人认证眼纹版开发文档

产品特点

- 权威数据源：采用权威认证的数据源做比对，覆盖率接近100%。
- 高准确性：金融级精准度，真实场景中的误识率低于十万分之一。
- 高安全性：全球独有眼纹专利和领先活体检测技术，能有效拦截照片、视频、3D 软件模拟及面具攻击。
- 海量业务验证：经实践验证，成熟可靠，服务超过 2 亿互联网金融用户，保障超 20 亿次交易安全。
- 极致体验：全球首创非配合式活体检测，无需用户做任何动作，无需注册采集，秒级验证通过。
- 高稳定性：基于蚂蚁金融云高可用、动态扩展的服务框架体系。

无线SDK+服务端接入

时序图



1. 接入方客户端初始化无线认证 SDK。
2. 接入方客户端调用无线认证 SDK 相关接口获取metaInfo。
3. 无线认证 SDK 将metaInfo返回给接入方客户端。
4. 接入方客户端将metaInfo上传给接入方服务端。
5. 接入方服务端访问实人认证服务端发起认证请求。
6. 实人认证服务端将queryId返回给接入方服务端。
7. 接入方服务端将queryId返回给接入方客户端。
8. 接入方客户端调用无线认证 SDK，开始刷脸认证。
9. 无线认证 SDK 完成刷脸认证并退出。
10. 接入方客户端通知接入方服务端去获取认证结果。
11. 接入方服务端访问实人认证服务端查询认证结果。
12. 实人认证服务端将认证结果返回给接入方服务端。
13. 接入方服务端处理认证结果，并通过接入方客户端向用户展示结果。

服务端文档

- 调用方式

接口基本信息：

- 服务请求地址：<https://saf.cn-shanghai.aliyuncs.com>
- 协议：HTTPS
- 方式：POST

- 调用参数

— 请求参数

名称	类型	是否必需	描述
Format	String	是	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，本版本对应为2017-03-31。
AccessKeyId	String	是	阿里云颁发给用户的访问服务所用的密钥ID，请参见 创建AccessKey 。
Signature	String	是	签名结果串，关于签名的计算方法，请参见 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。格式为：YYYY-MM-DDThh:mm:ssZ。 例如，2014-7-29T12:00:00Z（为北京时间2014年7月29日的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。
Action	String	是	取值必须是ExecuteRequest。
Service	String	是	具体服务名，本服务该参数固定为face_verify。
ServiceParameters	String	是	具体服务参数的json格式串，具体定义见下面各接口参数定义。

— 统一返回Code



说明：

文档中的返回code做了合并处理，事实上各个环节都有明确的返回码解析，可以暴露出来便于结果查询，也可以统一封装成以上统一code。暂时按照统一封装code返回，减少用户接入成本。

Code	Message
200	请求正常。
400	参数ServiceParameters不合法。
402	日QPS超过已购规格，限流。
403	权限不足，服务未开通或已到期。
404	参数Service不合法。
500	内部服务器错误。

- 发起认证请求

- 请求参数

发起认证请求时，其他参数参见[调用参数](#)，并在ServiceParameters的json字符串里传入以下字段：

名称	类型	是否必需	描述	示例值
method	String	是	取值必须是init。	init
certNumber	String	是	用户身份证件号。	33010320180101010X
name	String	是	用户姓名。	张三
bizId	String	是	接入方请求的唯一标识，由字母、数字、下划线组成，商户须确保其唯一性。 建议：开头使用一段自定义商户简称，中间使用一段日期，末尾使用一个序列。	ZSTP2018013076546767654695203625
metainfo	String	是	metainfo环境参数，需要通过客户端 SDK 获取（详见 iOS接入 和 Android接入 ）。	{"appName": "... }"

- 响应参数

名称	类型	是否必需	描述	示例值
queryId	String	是	刷脸认证唯一标识。如果初始化失败则为空，可通过返回码分析具体原因。	731be7f204a962b0486a9b64ea3050ae9b64ea3050ae

— 响应成功示例

```
{
  "Data": {
    "queryId": "731be7f204a962b0486a9b64ea3050ae",
    "bizId": "ZSTP2018013076546767654695203625"
  },
  "Message": "OK",
  "Code": 200,
  "RequestId": "4EF9AEE5-288F-4176-8110-00EDC91A614E"
}
```

— 响应失败示例

```
{
  "Message": "服务器内部错误",
  "Code": 500,
  "RequestId": "4EF9AEE5-288F-4176-8110-00EDC91A614E"
}
```

或者

```
{
  "Message": "name不能为空",
  "Code": 400,
  "RequestId": "4EF9AEE5-288F-4176-8110-00EDC91A614E"
}
```

— 返回码说明

见[统一返回码](#)所述，如客户选择该环节详细返回码透出，明细返回码如下：

resultCode（返回码）	resultCodeSub（明细返回码）	resultMsgSub（明细返回码对应的文案）	说明
OK	Z8100	初始化成功（Z8100）	初始化成功。
INVALID_PARAMETER	Z8101	抱歉，系统出错了，请您稍后再试（Z8101）	初始化时传入参数不正确。

resultCode (返回码)	resultCodeSub (明细返回码)	resultMsgSub (明细返回码对应的文案)	说明
INVALID_PARAMETER	Z8105	用户身份信息无效，请填写正确的身份信息 (Z8105)	请求参数中的用户身份信息无效，请填写正确的身份信息。
PRODUCT_NOT_OPEN	Z8102	请去开通产品 (Z8102)	当前租户未开通刷脸认证产品。
SYSTEM_ERROR	Z8199	抱歉，系统出错了，请稍后再试 (Z8199)	初始化时发生系统错误。
DEVICE_NOT_SUPPORT	Z1108	当前设备不支持刷脸 (Z1108)	设备类型不支持。
SDKVERSION_NOT_SUPPORT	Z1110	应用版本过低 (Z1110)	SDK版本不支持。
OS_NOT_SUPPORT	Z1109	当前系统不支持刷脸 (Z1109)	系统版本不支持。
UNABLE_GET_IMAGE	Z1102	抱歉，您暂时无法使用刷脸服务 (Z1102)	比对源不可用。
SYSTEM_ERROR	Z1111	抱歉，系统出错了，请您稍后再试 (Z1111)	初始化策略结果获取失败。
SYSTEM_ERROR	Z1112	抱歉，系统出错了，请您稍后再试 (Z1112)	暂时无法使用刷脸认证。
SYSTEM_ERROR	Z1199	抱歉，系统出错了，请您稍后再试 (Z1199)	暂时无法使用刷脸认证。
INVALID_PARAMETER	Z5101	抱歉，系统出错了，请您稍后再试 (Z5101)	暂时无法使用刷脸认证。
INVALID_PARAMETER	Z5102	抱歉，系统出错了，请您稍后再试 (Z5102)	初始化时发生系统错误。

resultCode (返回码)	resultCodeSub (明细返回码)	resultMsgSub (明细返回码对应的文案)	说明
INVALID_PARAMETER	Z5103	抱歉，系统出错了，请您稍后再试 (Z5103)	初始化时发生系统错误。
SYSTEM_ERROR	Z5199	抱歉，系统出错了，请您稍后再试 (Z5199)	初始化时发生系统错误。
HIGH_RISK	Z1114	抱歉，系统出错了，请您稍后再试 (Z1114)	刷脸失败次数过多，请您稍后再试。

- 查询认证结果

- 请求参数

查询认证结果时，其他参数参见[调用参数](#)，并在 **ServiceParameters** 的 json 字符串里传入以下字段：

名称	类型	是否必需	描述	示例值
method	String	是	表示是来查询人脸比对结果的，值必须为query。	query
bizId	String	是	接入方请求的唯一标识，与发起认证请求时传入的 bizId 保持一致。	ZSTP2018013076546767654695203625
queryId	String	是	刷脸认证唯一标识，来自 init 接口的结果。	731be7f204a962b0486a9b64ea3050ae
returnImage	String	否	刷脸认证的人脸活体照片。	"1" 或者 1 (返回采集的照片的base64结构)

- 响应参数

名称	类型	是否必需	描述	示例值
image	String	否	当入参含 returnImage 时，返回人脸认证SDK采集的人脸照片base64编码。	_9j_4AAQSkZJRgABAQAA SABIAAD....

— 响应成功示例

当入参不含returnImage时

```
{
  "Message": "OK",
  "Code": 200,
  "RequestId": "4EF9AEE5-288F-4176-8110-00EDC91A614E"
}
```

当入参含returnImage时

```
{
  "Message": "OK",
  "Code": 200,
  "Data": {
    "image": "_9j_4AAQSkZJRgABAQAA  
SABIAAD...."
  },
  "RequestId": "4EF9AEE5-288F-4176-8110-00EDC91A614E"
}
```

— 响应失败示例

```
{
  "Message": "不通过的原因",
  "Code": 400,
  "RequestId": "4EF9AEE5-288F-4176-8110-00EDC91A614E"
}
```

或者

```
{
  "Message": "服务器内部错误",
  "Code": 500,
  "RequestId": "4EF9AEE5-288F-4176-8110-00EDC91A614E"
}
```

— 返回码说明

见[统一返回码](#)所述，如客户选择该环节详细返回码透出，明细返回码如下：

resultCode (返回码)	resultCodeSub (明细返回码)	resultMsgSub (明细返回码对应的文案)	说明
OK	Z8300	认证成功 (Z8300)	刷脸认证通过。
INVALID_PARAMETER	Z8301	抱歉，系统出错了，请您稍后再试 (Z8301)	查询时传入参数不正确。
NOT_SAME_PERSON	Z1146	抱歉，没有认出您 (Z1146)	刷脸认证未通过。
PRODUCT_NOT_OPEN	Z8302	请去开通产品 (Z8302)	当前租户未开通刷脸认证产品。
SYSTEM_ERROR	Z8399	抱歉，系统出错了，请稍后再试 (Z8399)	查询时发生系统错误。
PROCESSING	Z5137	抱歉，系统出错了，请稍后再试 (Z5137)	刷脸认证未完成。

- 完整代码示例 (**Java**语言)

```
import java.io.UnsupportedEncodingException;
import java.net.URLEncoder;
import java.text.SimpleDateFormat;
import java.util.Arrays;
import java.util.Date;
import java.util.HashMap;
import java.util.Map;
import java.util.SimpleTimeZone;
import java.util.UUID;

import org.apache.commons.io.FileUtils;

import com.google.common.io.BaseEncoding;

public class PopTestV2 {
    public static void main(String[] args) throws Throwable {
        saf_pop_test();
    }

    public static void saf_pop_test() throws Throwable {
        //阿里云账号的 appKey 和 appSecret
        String appKey = "*****";
        String appSecret = "*****";
        String safUrl = "https://saf.cn-shanghai.aliyuncs.com";

        Map<String, String> serviceParameters = tools.MapUtil.create
();
```

```

//method value:init,query,match
serviceParameters.put("method", "init");
serviceParameters.put("bizid", "15888827666");
serviceParameters.put("name", "张三");
serviceParameters.put("certNumber", "33010320180101010X");
serviceParameters.put("metainfo", "*****");

/*method = query
serviceParameters.put("method", "query");
serviceParameters.put("bizid", "15888827666");
serviceParameters.put("queryId", "15888827666");
serviceParameters.put("returnImage", "1");

//method = match
serviceParameters.put("method", "match");
serviceParameters.put("name", "张三");
serviceParameters.put("certNumber", "33010320180101010X");
serviceParameters.put("imgbase64", "*****");
*/

// -----继续追加其他参数，不同的service，需要的参数也会不
同
// serviceParameters.put...
saf(appKey, appSecret, safUrl, "2017-03-31", "face_verify",
serviceParameters);
}

public static String doPOPRequest(String appKey, String
appSecret, String url, String action, String v,
                                Map<String, String> parameters
)
    throws Throwable {
    // 本来依赖fastjson和http client包
    // 阿里云账号的AK，需替换成你们自己的线上账号AK。
    // https://help.aliyun.com/document_detail/66453.html?spm=
a2c4g.11186623.6.547.PovY4k
    String encoding = "UTF-8";
    Map<String, String> paramsForPOP = new HashMap<>();

    // 公共参数-固定
    paramsForPOP.put("AccessKeyId", appKey);
    paramsForPOP.put("Format", "JSON");// 固定
    paramsForPOP.put("SignatureMethod", "HMAC-SHA1");// 固定
    paramsForPOP.put("Timestamp", formatIso8601Date(new Date
    ( )));// 注意格式是：YYYY-MM-DDThh:mm:ssZ
    paramsForPOP.put("SignatureVersion", "1.0");// 固定
    paramsForPOP.put("SignatureNonce", UUID.randomUUID().
toString());// 自行生成一个随机串，每次请求不可重复
    paramsForPOP.put("Version", v);// 固定
    paramsForPOP.put("Action", action);// 固定 -->"ExecuteRequest
    "

    if (parameters != null) {
        parameters.remove("Signature");
        paramsForPOP.putAll(parameters);
    }

    // 得到签名 https://www.alibabacloud.com/help/zh/doc-detail/
27572.htm?spm=a3c0i.o27570zh.b99.80.7c7ad9cd6jC8dM

```

```

        // https://help.aliyun.com/document_detail/66384.html?spm=a2c4g.11186623.6.543.wIP2rf
        String signature = computeSignature(paramsForPOP, appSecret, encoding);
        paramsForPOP.put("Signature", signature); // 把签名附加到post参数里

        // 发送请求, 可以替换为自己的http request lib
        String result = tools.http.HttpRequestBuilder.create(url).data(paramsForPOP).readTimeout(8500).charset(encoding).post().toString();

        // --->>> {"Data":{"score":56.0}, "Message":"OK", "Code":200}

        System.out.println(result);
        return result;
    }

    public static String saf(String appKey, String appSecret, String url, String v, String service,
                             Map<String, String> serviceParameters)
        throws Throwable {

        Map<String, String> params = new HashMap<>();

        // 接口业务参数-固定
        params.put("Service", service); // 服务code, 必填, 由阿里云颁发, 不可随意, account_abuse=垃圾账户识别; coupon_abuse=活动作弊 ...

        // 附上业务详细参数, 服务详细参数, 具体见文档里的业务参数部分, 业务参数整体转换为json格式
        params.put("ServiceParameters", com.alibaba.fastjson.JSON.toJSONString(serviceParameters));

        return doPOPRequest(appKey, appSecret, url, "ExecuteRequest", v, params);
    }

    private static String formatIso8601Date(Date date) {
        SimpleDateFormat df = new SimpleDateFormat("yyyy-MM-dd'T'HH:mm:ss'Z'");
        df.setTimeZone(new SimpleTimeZone(0, "GMT")); // 注意使用GMT时间
        return df.format(date);
    }

    public static String computeSignature(Map<String, String> parameters, String secret, String encoding)
        throws Exception {
        // 将参数Key按字典顺序排序
        String[] sortedKeys = parameters.keySet().toArray(new String[] {});
        Arrays.sort(sortedKeys);

        String separator = "&";
        boolean first = true;

        // 生成规范化请求字符串
        StringBuilder sb = new StringBuilder();
        for (String key : sortedKeys) {

```

```
        if (first) {
            first = false;
        } else {
            sb.append("&");
        }

        sb.append(encode(key, encoding)).append("=").append(
            encode(parameters.get(key), encoding));
    }

    // 生成用于计算签名的字符串 toSign
    StringBuilder toSign = new StringBuilder();
    toSign.append("POST").append(separator);
    toSign.append(encode("/", encoding)).append(separator);
    toSign.append(encode(sb.toString(), encoding));

    // System.out.println("==> " + toSign.toString());

    // 注意点：在secret后面要加入一个字符"&"
    return getSignature("HmacSHA1", (secret + separator).
        getBytes(encoding), toSign.toString().getBytes(encoding));
}

private static String encode(String value, String encoding)
throws UnsupportedOperationException {
    if (value == null)
        return null;

    // 使用URLCoder.encode编码后，将"+", "*", "%7E"做替换即满足 API规
    定的编码规范
    return URLEncoder.encode(value, encoding).replace("+", "%20")
        .replace("*", "%2A").replace("%7E", "~");
}

public static String getSignature(String algorithm, byte[]
secret, byte[] data) {
    // return tools.Convert.getSignature(algorithm, secret, data
    );
    try {
        // String algorithm = "HmacSHA384";//HmacSHA256";
        javax.crypto.Mac mac = javax.crypto.Mac.getInstance(
            algorithm);
        mac.init(new javax.crypto.spec.SecretKeySpec(secret,
            algorithm));

        return BaseEncoding.base64().encode(mac.doFinal(data));
    } catch (Exception e) { // InvalidKeyException NoSuchAlgo
        rithmException
        e.printStackTrace();
    }

    return null;
}
}
```

客户端文档

实人认证提供客户端 SDK 帮助接入方在 APP 中实现刷脸认证功能。接入方可通过服务端调用接口 [发起认证请求](#) 获取刷脸认证唯一标识 **queryId**，并使用 **queryId** 唤起无线认证 SDK 进行刷脸认证。

iOS接入

1. 下载 iOS SDK 及 Demo 工程。

- iOS SDK : [FaceVerifySDK_iOS_3.0.zip](#)
- iOS Demo : [FaceDetectDemo-iOS-master-20181121.zip](#)

2. 添加 SDK。

在 **Apple Xcode** 中，选择 **TARGETS**，单击 **General** 标签页，展开 **Linked Frameworks and Libraries** 列表，添加以下 Framework：

- *APBToygerFacade.framework*
- *ZolozMobileRPC.framework*
- *APPSecuritySDK.framework*
- *BioAuthAPI.framework*
- *BioAuthEngine.framework*
- *MPRemoteLogging.framework*
- *ToygerService.framework*
- *ZolozIdentityManager.framework*
- *ZolozSensorServices.framework*
- *ZolozOpenPlatformBuild.framework*

Name	Status
 APBToygerFacade.framework	Required 
 ToygerService.framework	Required 
 APPSecuritySDK.framework	Required 
 ZolozMobileRPC.framework	Required 
 ZolozIdentityManager.framework	Required 
 BioAuthAPI.framework	Required 
 ZolozSensorServices.framework	Required 
 BioAuthEngine.framework	Required 
 ZolozOpenPlatformBuild.framework	Required 
 MPRemoteLogging.framework	Required 

3. 添加系统库。

在 **Linked Frameworks and Libraries** 中添加如下 iOS 系统库：

- *CoreGraphics.framework*
- *Accelerate.framework*
- *SystemConfiguration.framework*
- *AssetsLibrary.framework*
- *CoreTelephony.framework*
- *QuartzCore.framework*
- *CoreFoundation.framework*
- *CoreLocation.framework*
- *ImageIO.framework*
- *CoreMedia.framework*
- *CoreMotion.framework*
- *AVFoundation.framework*
- *libz.tbd*
- *libc++.1.tbd*
- *libstdc++.6.0.9.tbd*
- *libc++abi.tbd*

 CoreGraphics.framework	Required 
 Accelerate.framework	Required 
 SystemConfiguration.framework	Required 
 AssetsLibrary.framework	Required 
 CoreTelephony.framework	Required 
 QuartzCore.framework	Required 
 CoreFoundation.framework	Required 
 CoreLocation.framework	Required 
 ImageIO.framework	Required 
 CoreMedia.framework	Required 
 CoreMotion.framework	Required 
 AVFoundation.framework	Required 
 libz.tbd	Required 
 libc++.1.tbd	Required 
 libc++abi.tbd	Required 
 libstdc++.6.0.9.tbd	Required 

4. 拷贝资源文件。

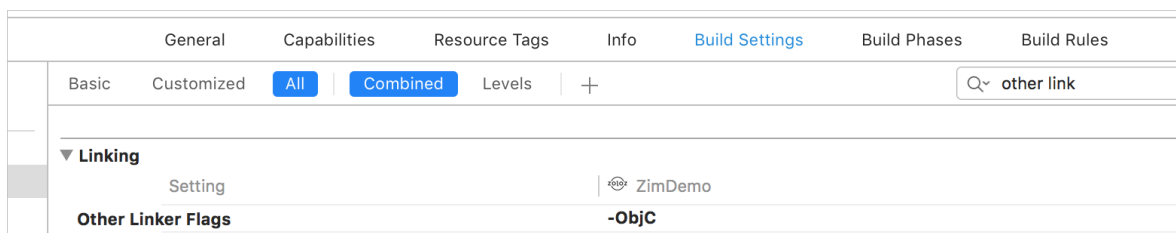
选择 **TARGETS**，单击 **Build Phases** 标签页，在 **Copy Bundle Resources** 中添加如下三个 bundle：

- *APBToygerFacade.bundle*：位于 *APBToygerFacade.framework* 中
- *ToygerService.bundle*：位于 *ToygerService.framework* 中
- *BioAuthEngine.bundle*：位于 *BioAuthEngine.framework* 中

 APBToygerFacade.bundle ...in ZimDemo/frameworks/APBToygerFacade.framework
 ToygerService.bundle ...in ZimDemo/frameworks/ToygerService.framework
 BioAuthEngine.bundle ...in ZimDemo/frameworks/BioAuthEngine.framework

5. 链接器参数设置。

选择 **TARGETS**，单击 **Build Settings** 标签页，在 **Linking** 中设置 **Other linker flags** 值为 -ObjC。



6. 调用 SDK。

- 引入头文件。

```
#import <ZolozIdentityManager/ZolozIdentityPublicApi.h>
```

- 初始化 SDK。

```
[ZolozSdk init]
```

初始化接口是异步执行，不会影响主线程 UI 渲染，建议接入方将初始化接口放在 `AppDelegate` 的如下函数中调用：

```
(BOOL)application:(UIApplication)applicationDidFinishLaunchingWithOptions:(NSDictionary)launchOptions
```

- 获取 **metainfo**。

```
[ZolozIdentityManagergetMetaInfo]
```

返回值是 `NSDictionary`，接入方服务端需要将其转换为 JSON 字符串，调用实人认证服务端 [发起认证请求](#) 时需要传入 **metainfo**。

- 开始刷脸认证。

```
[[ZolozIdentityManager sharedInstance] verifyWith:_queryId  
extParams:extParams onCompletion:^(ZIMResponse *response) {  
    NSString *title = @"刷脸成功";  
    switch (response.code) {  
        case 1000:  
            break;  
        case 1001:  
            title = @"系统错误";  
            break;  
        case 1003:  
            title = @"验证中断";  
            break;  
        case 2002:  
            title = @"网络错误";  
            break;  
        case 2006:  
            title = @"刷脸失败";  
            break;  
        default:  
            break;  
    }  
}
```



```
}
```

其中，

- 入参第一个是 **queryId**，通过服务端 [发起认证请求](#) 获取（在 SDK 接口声明和 Demo 工程中此参数名为 **zimId**，含义与 **queryId** 相同）
- 入参第二个是 **extParams**，具体为：

名称	类型	是否必需	描述
currentCtr	对象	是	当前页面的controller。

- response.code**类型是整型，有以下五种返回值，：

Code	说明
1000	刷脸成功，该结果仅供参考，可通过服务端人脸结果查询获取最终认证结果。
1001	系统错误。
1003	验证中断。
2002	网络错误。
2006	刷脸失败，如需获取更详细的失败原因，需调用服务端人脸结果查询接口。

Android接入

1. 下载 Android SDK 及 Demo 工程。

- Android SDK：[FaceVerifySDK_Android_3.1.zip](#)
- Android Demo：[FaceDetectDemo-Android-master-20181121.zip](#)



说明：

此 Android Demo 采用的是通过 gradle 拉取 maven 仓库下载 SDK，您可以修改为直接引入 SDK 的方式。

2. 添加 SDK。

将下载的 SDK 解压后复制到项目下的 **libs** 目录（如果不存在 **libs** 目录，请自行创建），并在 **build.gradle** 文件中，增加以下依赖：

```
dependencies {
    ...
    compile 'com.android.support:support-v4:26.0.0-alpha1'
```

```
compile files('libs/deviceid-release-6.0.2.20181008.aar')
compile files('libs/fastjson-1.1.45.jar')
compile files('libs/sdk_antcloud-3.0.0.31.aar')
}
```

3. 调用 SDK。

- 初始化 SDK。

```
ZIMFacade.install(context);
```

- 获取 **metainfo**。

```
String metaInfos = ZIMFacade.getMetaInfos(context);
```

接入方服务端在调用实人认证服务端 [发起认证请求](#) 时需要传入 **metainfo**。

- 开始刷脸认证。

```
ZIMFacade zimFacade = ZIMFacadeBuilder.create(MainActivity.this);
zimFacade.verify(queryId, null, new ZIMCallback() {
    @Override
    public boolean response(ZIMResponse response) {
        // TODO: 接入方在这里实现自身业务
        if (null != response && 1000 == response.code) {
            // 认证成功
        } else {
            // 认证失败
        }
        return true;
    }
})
```

其中，

- 入参第一个是 **queryId**，通过服务端 [发起认证请求](#) 获取（在 SDK 接口声明和 Demo 工程中此参数名为 zimId，含义与 queryId 相同）
- response.code** 类型是整型，有以下五种返回值：

Code	说明
1000	刷脸成功，该结果仅供参考，可通过服务端人脸结果查询获取最终认证结果。
1001	系统错误。
1003	验证中断。
2002	网络错误。
2006	刷脸失败，如需获取更详细的失败原因，需调用服务端人脸结果查询接口。

纯服务端接入

请求参数

纯服务端接入实人认证时，调用接口的方式和参数参见 [无线SDK+服务端接入 - 服务端文档](#) 中 [调用方式](#) 和 [调用参数](#) 章节，并在 **ServiceParameters** 的 json 字符串里传入以下字段：

名称	类型	是否必需	描述
method	String	是	值必须为match。
name	String	是	用户姓名。
certNumber	String	是	用户身份证件号。
imgbase64	String	是	人脸图片二进制数据base64编码。

返回码说明

见[统一返回码](#)所述。