本文档强烈推荐使用Web版式视图阅读

在菜单栏点击“视图”🡪“Web版式视图”完成设置

1.人脸检测与属性分析

**人脸识别接口分为V2和V3两个版本，本文档为V3版本接口的说明文档，请确认您在百度云后台获得的是V3版本接口权限，再来阅读本文档。**

辨别接口版本的方法是：在百度云后台进入【应用列表】，点击【应用名称】，在【API列表】中可以看到【请求地址】，若请求地址中带有【v3】标识，则您具有的是v3权限，可以阅读本文档；若请求地址中带有【v2】标识，则您具有的是v2权限，应该去阅读v2文档。 如果您对文档内容有任何疑问，可以通过以下几种方式联系我们：

* 在百度云控制台内 [提交工单](http://ticket.bce.baidu.com/#/ticket/create)，咨询问题类型请选择**人工智能服务**；
* 如有需要讨论的疑问，欢迎进入 [AI社区](http://ai.baidu.com/forum/topic/list/165) 与其他开发者们一同交流。

1.1.能力介绍

**接口能力**

* **人脸检测**：检测图片中的人脸并标记出位置信息;
* **人脸关键点**：展示人脸的核心关键点信息，及72个关键点信息。
* **人脸属性值**：展示人脸属性信息，如年龄、性别等。
* **人脸质量信息**：返回人脸各部分的遮挡、光照、模糊、完整度、置信度等信息。

**业务应用**

典型应用场景：如**人脸属性分析**，**基于人脸关键点的加工分析**，**人脸营销活动**等。

说明：检测响应速度，与图片中人脸数量相关，人脸数量较多时响应时间会有些许延长。

**质量检测**

如果需要判断一张图片中的人脸，是否符合后续识别或者对比的条件，可以使用此接口，在请求时在face\_field参数中请求quality。基于返回结果quality中，以下字段及对应阈值，进行质量检测的判断，以保证人脸质量符合后续业务操作要求。

| 指标 | 字段与解释 | 推荐数值界限 |
| --- | --- | --- |
| 遮挡范围 | **occlusion**，取值范围[0~1]，0为无遮挡，1是完全遮挡 含有多个具体子字段，表示脸部多个部位 通常用作判断头发、墨镜、口罩等遮挡 | left\_eye : 0.6, #左眼被遮挡的阈值 right\_eye : 0.6, #右眼被遮挡的阈值 nose : 0.7, #鼻子被遮挡的阈值 mouth : 0.7, #嘴巴被遮挡的阈值 left\_cheek : 0.8, #左脸颊被遮挡的阈值 right\_cheek : 0.8, #右脸颊被遮挡的阈值 chin\_contour : 0.6, #下巴被遮挡阈值 |
| 模糊度范围 | **blur**，取值范围[0~1]，0是最清晰，1是最模糊 | 小于0.7 |
| 光照范围 | **illumination**，取值范围[0~255] 脸部光照的灰度值，0表示光照不好 以及对应客户端SDK中，YUV的Y分量 | 大于40 |
| 姿态角度 | **Pitch**：三维旋转之俯仰角度[-90(上), 90(下)] **Roll**：平面内旋转角[-180(逆时针), 180(顺时针)] **Yaw**：三维旋转之左右旋转角[-90(左), 90(右)] | 分别小于20度 |
| 人脸完整度 | **completeness**（0或1），0为人脸溢出图像边界， 1为人脸都在图像边界内 | 视业务逻辑判断 |
| 人脸大小 | 人脸部分的大小 建议长宽像素值范围：80\*80~200\*200 | 人脸部分不小于**100\*100**像素 |

1.2.调用方式

**请求URL数据格式**

向API服务地址使用POST发送请求，必须在URL中带上参数access\_token，可通过后台的API Key和Secret Key生成，具体方式请参考“[Access Token获取](http://ai.baidu.com/docs#/Auth)”。

**获取access\_token的示例代码**

Java

import org.json.JSONObject;

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.net.HttpURLConnection;

import java.net.URL;

import java.util.List;

import java.util.Map;

/\*\*

\* 获取token类

\*/

public class AuthService {

/\*\*

\* 获取权限token

\* @return 返回示例：

\* {

\* "access\_token": "24.460da4889caad24cccdb1fea17221975.2592000.1491995545.282335-1234567",

\* "expires\_in": 2592000

\* }

\*/

public static String getAuth() {

// 官网获取的 API Key 更新为你注册的

String clientId = "百度云应用的AK";

// 官网获取的 Secret Key 更新为你注册的

String clientSecret = "百度云应用的SK";

return getAuth(clientId, clientSecret);

}

/\*\*

\* 获取API访问token

\* 该token有一定的有效期，需要自行管理，当失效时需重新获取.

\* @param ak - 百度云官网获取的 API Key

\* @param sk - 百度云官网获取的 Securet Key

\* @return assess\_token 示例：

\* "24.460da4889caad24cccdb1fea17221975.2592000.1491995545.282335-1234567"

\*/

public static String getAuth(String ak, String sk) {

// 获取token地址

String authHost = "https://aip.baidubce.com/oauth/2.0/token?";

String getAccessTokenUrl = authHost

// 1. grant\_type为固定参数

+ "grant\_type=client\_credentials"

// 2. 官网获取的 API Key

+ "&client\_id=" + ak

// 3. 官网获取的 Secret Key

+ "&client\_secret=" + sk;

try {

URL realUrl = new URL(getAccessTokenUrl);

// 打开和URL之间的连接

HttpURLConnection connection = (HttpURLConnection) realUrl.openConnection();

connection.setRequestMethod("GET");

connection.connect();

// 获取所有响应头字段

Map<String, List<String>> map = connection.getHeaderFields();

// 遍历所有的响应头字段

for (String key : map.keySet()) {

System.err.println(key + "--->" + map.get(key));

}

// 定义 BufferedReader输入流来读取URL的响应

BufferedReader in = new BufferedReader(new InputStreamReader(connection.getInputStream()));

String result = "";

String line;

while ((line = in.readLine()) != null) {

result += line;

}

/\*\*

\* 返回结果示例

\*/

System.err.println("result:" + result);

JSONObject jsonObject = new JSONObject(result);

String access\_token = jsonObject.getString("access\_token");

return access\_token;

} catch (Exception e) {

System.err.printf("获取token失败！");

e.printStackTrace(System.err);

}

return null;

}

}

注意：access\_token的有效期为30天，切记需要每30天进行定期更换，或者每次请求都拉取新token；

例如此接口，使用HTTPS POST发送：

https://aip.baidubce.com/rest/2.0/face/v3/detect?access\_token=24.f9ba9c5341b67688ab5added8bc91dec.2592000.1485570332.282335-8574074

POST中Body的参数，按照下方请求参数说明选择即可。

提示：如果您为百度云老用户，正在使用其他非AI的服务，可以参考[百度云AKSK鉴权方式](https://cloud.baidu.com/doc/Reference/AuthenticationMechanism.html)发送请求，虽然请求方式和鉴权方法和本文所介绍的不同，但请求参数和返回结果一致。

1.3.请求说明

**注意事项**：

* **请求体格式化**：Content-Type为application/json，通过json格式化请求体。
* **Base64编码**：请求的图片需经过Base64编码，图片的base64编码指将图片数据编码成一串字符串，使用该字符串代替图像地址。您可以首先得到图片的二进制，然后用Base64格式编码即可。需要注意的是，图片的base64编码是不包含图片头的，如data:image/jpg;base64,
* **图片格式**：现支持PNG、JPG、JPEG、BMP，**不支持GIF图片**
* **人脸识别接口分为V2和V3两个版本，本文档为V3版本接口的说明文档，请确认您在百度云后台获得的是V3版本接口权限，再来阅读本文档。**

辨别接口版本的方法是：在百度云后台进入【应用列表】，点击【应用名称】，在【API列表】中可以看到【请求地址】，若请求地址中带有【v3】标识，则您具有的是v3权限，可以阅读本文档；若请求地址中带有【v2】标识，则您具有的是v2权限，应该去阅读v2文档。

**请求示例**

HTTP方法：POST

请求URL： https://aip.baidubce.com/rest/2.0/face/v3/detect

URL参数：

| 参数 | 值 |
| --- | --- |
| access\_token | 通过API Key和Secret Key获取的access\_token,参考“[Access Token获取](http://ai.baidu.com/docs#Beginner-Auth)” |

Header：

| 参数 | 值 |
| --- | --- |
| Content-Type | application/json |

Body中放置请求参数，参数详情如下：

**请求参数**

| 参数 | 必选 | 类型 | 说明 |
| --- | --- | --- | --- |
| image | 是 | string | 图片信息(**总数据大小应小于10M**)，图片上传方式根据image\_type来判断 |
| image\_type | 是 | string | 图片类型 **BASE64**:图片的base64值，base64编码后的图片数据，编码后的图片大小不超过2M； **URL**:图片的 URL地址( 可能由于网络等原因导致下载图片时间过长)； **FACE\_TOKEN**: 人脸图片的唯一标识，调用人脸检测接口时，会为每个人脸图片赋予一个唯一的FACE\_TOKEN，同一张图片多次检测得到的FACE\_TOKEN是同一个。 |
| face\_field | 否 | string | 包括**age,beauty,expression,faceshape,gender,glasses,landmark,race,quality,facetype信息** 逗号分隔. 默认只返回face\_token、人脸框、概率和旋转角度 |
| max\_face\_num | 否 | uint32 | 最多处理人脸的数目，默认值为1，仅检测图片中面积最大的那个人脸；**最大值10**，检测图片中面积最大的几张人脸。 |
| face\_type | 否 | string | 人脸的类型 **LIVE**表示生活照：通常为手机、相机拍摄的人像图片、或从网络获取的人像图片等 **IDCARD**表示身份证芯片照：二代身份证内置芯片中的人像照片 **WATERMARK**表示带水印证件照：一般为带水印的小图，如公安网小图 **CERT**表示证件照片：如拍摄的身份证、工卡、护照、学生证等证件图片 默认**LIVE** |

说明：face\_field参数，默认只返回人脸框、概率和旋转角度，age等更多属性，请在此参数中添加。

**请求示例代码**

**提示一**：使用示例代码前，请记得替换其中的示例Token、图片地址或Base64信息。

**提示二**：部分语言依赖的类或库，请在代码注释中查看下载地址。

Java

package com.baidu.ai.aip;

import com.baidu.ai.aip.utils.HttpUtil;

import com.baidu.ai.aip.utils.GsonUtils;

import java.util.\*;

/\*\*

\* 人脸检测与属性分析

\*/

public class FaceDetect {

/\*\*

\* 重要提示代码中所需工具类

\* FileUtil,Base64Util,HttpUtil,GsonUtils请从

\* https://ai.baidu.com/file/658A35ABAB2D404FBF903F64D47C1F72

\* https://ai.baidu.com/file/C8D81F3301E24D2892968F09AE1AD6E2

\* https://ai.baidu.com/file/544D677F5D4E4F17B4122FBD60DB82B3

\* https://ai.baidu.com/file/470B3ACCA3FE43788B5A963BF0B625F3

\* 下载

\*/

public static String detect() {

// 请求url

String url = "https://aip.baidubce.com/rest/2.0/face/v3/detect";

try {

Map<String, Object> map = new HashMap<>();

map.put("image", "027d8308a2ec665acb1bdf63e513bcb9");

map.put("face\_field", "faceshape,facetype");

map.put("image\_type", "FACE\_TOKEN");

String param = GsonUtils.toJson(map);

// 注意这里仅为了简化编码每一次请求都去获取access\_token，线上环境access\_token有过期时间， 客户端可自行缓存，过期后重新获取。

String accessToken = "[调用鉴权接口获取的token]";

String result = HttpUtil.post(url, accessToken, "application/json", param);

System.out.println(result);

return result;

} catch (Exception e) {

e.printStackTrace();

}

return null;

}

public static void main(String[] args) {

FaceDetect.detect();

}

}

1.4.返回说明

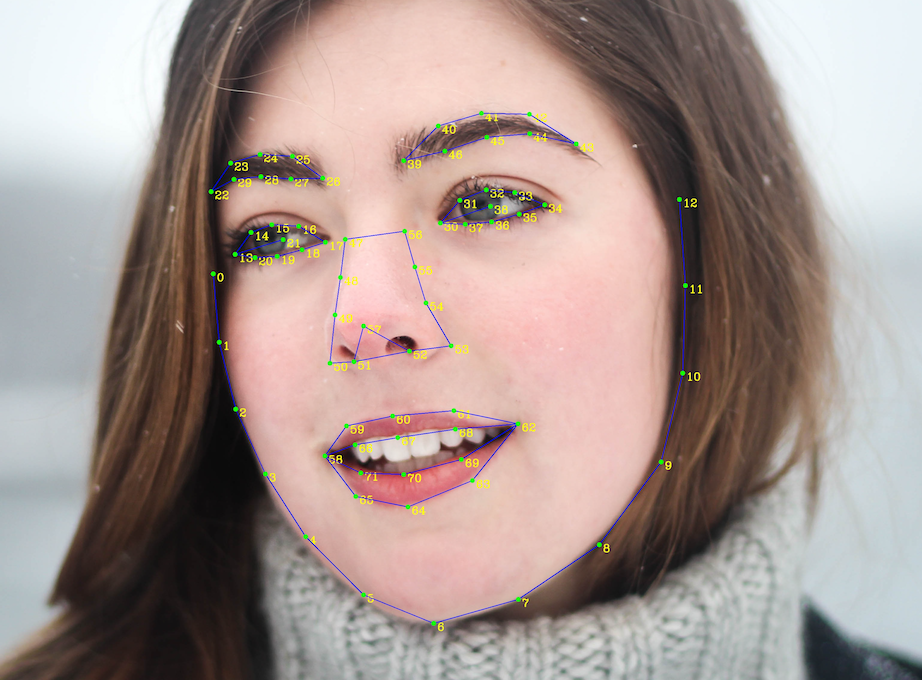
**返回参数**

* 返回结果

| 字段 | 必选 | 类型 | 说明 |
| --- | --- | --- | --- |
| face\_num | 是 | int | 检测到的图片中的人脸数量 |
| face\_list | 是 | array | 人脸信息列表，具体包含的参数参考下面的列表。 |
| +face\_token | 是 | string | 人脸图片的唯一标识 |
| +location | 是 | array | 人脸在图片中的位置 |
| ++left | 是 | double | 人脸区域离左边界的距离 |
| ++top | 是 | double | 人脸区域离上边界的距离 |
| ++width | 是 | double | 人脸区域的宽度 |
| ++height | 是 | double | 人脸区域的高度 |
| ++rotation | 是 | int64 | 人脸框相对于竖直方向的顺时针旋转角，[-180,180] |
| +face\_probability | 是 | double | 人脸置信度，范围【0~1】，代表这是一张人脸的概率，0最小、1最大。 |
| +angel | 是 | array | 人脸旋转角度参数 |
| ++yaw | 是 | double | 三维旋转之左右旋转角[-90(左), 90(右)] |
| ++pitch | 是 | double | 三维旋转之俯仰角度[-90(上), 90(下)] |
| ++roll | 是 | double | 平面内旋转角[-180(逆时针), 180(顺时针)] |
| +age | 否 | double | 年龄 ，当**face\_field包含age时返回** |
| +beauty | 否 | int64 | 美丑打分，范围0-100，越大表示越美。当**face\_fields包含beauty时返回** |
| +expression | 否 | array | 表情，当 **face\_field包含expression时返回** |
| ++type | 否 | string | **none**:不笑；**smile**:微笑；**laugh**:大笑 |
| ++probability | 否 | double | 表情置信度，范围【0~1】，0最小、1最大。 |
| +face\_shape | 否 | array | 脸型，当**face\_field包含faceshape时返回** |
| ++type | 否 | double | **square**: 正方形 **triangle**:三角形 **oval**: 椭圆 **heart**: 心形 **round**: 圆形 |
| ++probability | 否 | double | 置信度，范围【0~1】，代表这是人脸形状判断正确的概率，0最小、1最大。 |
| +gender | 否 | array | 性别，**face\_field包含gender时返回** |
| ++type | 否 | string | male:**男性** female:**女性** |
| ++probability | 否 | double | 性别置信度，范围【0~1】，0代表概率最小、1代表最大。 |
| +glasses | 否 | array | 是否带眼镜，**face\_field包含glasses时返回** |
| ++type | 否 | string | **none**:无眼镜，**common**:普通眼镜，**sun**:墨镜 |
| ++probability | 否 | double | 眼镜置信度，范围【0~1】，0代表概率最小、1代表最大。 |
| +race | 否 | array | 人种 **face\_field包含race时返回** |
| ++type | 否 | string | **yellow**: 黄种人 **white**: 白种人 **black**:黑种人 **arabs**: **阿拉伯人** |
| ++probability | 否 | double | 人种置信度，范围【0~1】，0代表概率最小、1代表最大。 |
| +facetype | 否 | array | 真实人脸/卡通人脸 **face\_field包含facetype时返回** |
| ++type | 否 | string | **human**: 真实人脸 **cartoon**: 卡通人脸 |
| ++probability | 否 | double | 人脸类型判断正确的置信度，范围【0~1】，0代表概率最小、1代表最大。 |
| +landmark | 否 | array | 4个关键点位置，左眼中心、右眼中心、鼻尖、嘴中心。**face\_field包含landmark时返回** |
| +landmark72 | 否 | array | 72个特征点位置 **face\_field包含landmark时返回** |
| +quality | 否 | array | 人脸质量信息。**face\_field包含quality时返回** |
| ++occlusion | 否 | array | 人脸各部分遮挡的概率，范围[0~1]，0表示完整，1表示不完整 |
| +++left\_eye | 否 | double | 左眼遮挡比例，[0-1] ，1表示完全遮挡 |
| +++right\_eye | 否 | double | 右眼遮挡比例，[0-1] ， 1表示完全遮挡 |
| +++nose | 否 | double | 鼻子遮挡比例，[0-1] ， 1表示完全遮挡 |
| +++mouth | 否 | double | 嘴巴遮挡比例，[0-1] ， 1表示完全遮挡 |
| +++left\_cheek | 否 | double | 左脸颊遮挡比例，[0-1] ， 1表示完全遮挡 |
| +++right\_cheek | 否 | double | 右脸颊遮挡比例，[0-1] ， 1表示完全遮挡 |
| +++chin | 否 | double | 下巴遮挡比例，，[0-1] ， 1表示完全遮挡 |
| ++blur | 否 | double | 人脸模糊程度，范围[0~1]，0表示清晰，1表示模糊 |
| ++illumination | 否 | double | 取值范围在[0~255], 表示脸部区域的光照程度 越大表示光照越好 |
| ++completeness | 否 | int64 | 人脸完整度，0或1, 0为人脸溢出图像边界，1为人脸都在图像边界内 |

* 返回示例
* {
* "face\_num": 1,
* "face\_list": [
* {
* "face\_token": "35235asfas21421fakghktyfdgh68bio",
* "location": {
* "left": 117,
* "top": 131,
* "width": 172,
* "height": 170,
* "rotation": 4
* },
* "face\_probability": 1,
* "angle" :{
* "yaw" : -0.34859421849251
* "pitch" 1.9135693311691
* "roll" :2.3033397197723
* }
* "landmark": [
* {
* "x": 161.74819946289,
* "y": 163.30244445801
* },
* ...
* ],
* "landmark72": [
* {
* "x": 115.86531066895,
* "y": 170.0546875
* }，
* ...
* ],
* "age": 29.298097610474,
* "beauty": 55.128883361816,
* "expression": {
* "type": "smile",
* "probability" : 0.5543018579483
* },
* "gender": {
* "type": "male",
* "probability": 0.99979132413864
* },
* "glasses": {
* "type": "sun",
* "probability": 0.99999964237213
* },
* "race": {
* "type": "yellow",
* "probability": 0.99999976158142
* },
* "face\_shape": {
* "type": "triangle",
* "probability": 0.5543018579483
* }
* "quality": {
* "occlusion": {
* "left\_eye": 0,
* "right\_eye": 0,
* "nose": 0,
* "mouth": 0,
* "left\_cheek": 0.0064102564938366,
* "right\_cheek": 0.0057411273010075,
* "chin": 0
* },
* "blur": 1.1886881756684e-10,
* "illumination": 141,
* "completeness": 1
* }
* }
* ]
* }

**72个关键点分布图（对应landmark72个点的顺序，序号从0-71）**：



**错误码**  
请参考[人脸识别错误码](http://ai.baidu.com/docs#/Face-ErrorCode-V3/top)

# 2.人脸对比

**人脸识别接口分为V2和V3两个版本，本文档为V3版本接口的说明文档，请确认您在百度云后台获得的是V3版本接口权限，再来阅读本文档。**

辨别接口版本的方法是：在百度云后台进入【应用列表】，点击【应用名称】，在【API列表】中可以看到【请求地址】，若请求地址中带有【v3】标识，则您具有的是v3权限，可以阅读本文档；若请求地址中带有【v2】标识，则您具有的是v2权限，应该去阅读v2文档。 如果您对文档内容有任何疑问，可以通过以下几种方式联系我们：

* 在百度云控制台内 [提交工单](http://ticket.bce.baidu.com/#/ticket/create)，咨询问题类型请选择**人工智能服务**；
* 如有需要讨论的疑问，欢迎进入 [AI社区](http://ai.baidu.com/forum/topic/list/165) 与其他开发者们一同交流。

## 2.1.能力介绍

**接口能力**

* **两张人脸图片相似度对比**：比对两张图片中人脸的相似度，并返回相似度分值；
* **多种图片类型**：支持**生活照**、**证件照**、**身份证芯片照**、**带网纹照**四种类型的人脸对比；
* **活体检测控制**：基于图片中的破绽分析，判断其中的人脸是否为**二次翻拍**（举例：如用户A用手机拍摄了一张包含人脸的图片一，用户B翻拍了图片一得到了图片二，并用图片二伪造成用户A去进行识别操作，这种情况普遍发生在金融开户、实名认证等环节。）；
* **质量检测控制**：分析图片的中人脸的模糊度、角度、光照强度等特征，判断图片质量；

**业务应用**

用于比对多张图片中的人脸相似度并返回两两比对的得分，可用于判断两张脸是否是同一人的可能性大小。

典型应用场景：如**人证合一验证**，**用户认证**等，可与您现有的人脸库进行比对验证。

## 2.2.调用方式

**请求URL数据格式**

向API服务地址使用POST发送请求，必须在URL中带上参数access\_token，可通过后台的API Key和Secret Key生成，具体方式请参考“[Access Token获取](http://ai.baidu.com/docs#/Auth)”。

**获取access\_token的示例代码**

Java

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.net.HttpURLConnection;

import java.net.URL;

import java.util.List;

import java.util.Map;

/\*\*

\* 获取token类

\*/

public class AuthService {

/\*\*

\* 获取权限token

\* @return 返回示例：

\* {

\* "access\_token": "24.460da4889caad24cccdb1fea17221975.2592000.1491995545.282335-1234567",

\* "expires\_in": 2592000

\* }

\*/

public static String getAuth() {

// 官网获取的 API Key 更新为你注册的

String clientId = "百度云应用的AK";

// 官网获取的 Secret Key 更新为你注册的

String clientSecret = "百度云应用的SK";

return getAuth(clientId, clientSecret);

}

/\*\*

\* 获取API访问token

\* 该token有一定的有效期，需要自行管理，当失效时需重新获取.

\* @param ak - 百度云官网获取的 API Key

\* @param sk - 百度云官网获取的 Securet Key

\* @return assess\_token 示例：

\* "24.460da4889caad24cccdb1fea17221975.2592000.1491995545.282335-1234567"

\*/

public static String getAuth(String ak, String sk) {

// 获取token地址

String authHost = "https://aip.baidubce.com/oauth/2.0/token?";

String getAccessTokenUrl = authHost

// 1. grant\_type为固定参数

+ "grant\_type=client\_credentials"

// 2. 官网获取的 API Key

+ "&client\_id=" + ak

// 3. 官网获取的 Secret Key

+ "&client\_secret=" + sk;

try {

URL realUrl = new URL(getAccessTokenUrl);

// 打开和URL之间的连接

HttpURLConnection connection = (HttpURLConnection) realUrl.openConnection();

connection.setRequestMethod("GET");

connection.connect();

// 获取所有响应头字段

Map<String, List<String>> map = connection.getHeaderFields();

// 遍历所有的响应头字段

for (String key : map.keySet()) {

System.err.println(key + "--->" + map.get(key));

}

// 定义 BufferedReader输入流来读取URL的响应

BufferedReader in = new BufferedReader(new InputStreamReader(connection.getInputStream()));

String result = "";

String line;

while ((line = in.readLine()) != null) {

result += line;

}

/\*\*

\* 返回结果示例

\*/

System.err.println("result:" + result);

JSONObject jsonObject = new JSONObject(result);

String access\_token = jsonObject.getString("access\_token");

return access\_token;

} catch (Exception e) {

System.err.printf("获取token失败！");

e.printStackTrace(System.err);

}

return null;

}

}

**注意**：access\_token的有效期为30天，**切记需要每30天进行定期更换，或者每次请求都拉取新token**；

例如此接口，使用HTTPS POST发送：

https://aip.baidubce.com/rest/2.0/face/v3/match?access\_token=24.f9ba9c5341b67688ab5added8bc91dec.2592000.1485570332.282335-8574075

POST中Body的参数，按照下方请求参数说明选择即可。

**提示**：如果您为百度云老用户，正在使用其他非AI的服务，可以参考[百度云AKSK鉴权方式](https://cloud.baidu.com/doc/Reference/AuthenticationMechanism.html)发送请求，虽然**请求方式**和**鉴权方法**和本文所介绍的不同，但请求参数和返回结果一致。

## 2.3.请求说明

**注意事项**：

* **请求体格式化**：Content-Type为application/json，通过json格式化请求体。
* **Base64编码**：请求的图片需经过Base64编码，图片的base64编码指将图片数据编码成一串字符串，使用该字符串代替图像地址。您可以首先得到图片的二进制，然后用Base64格式编码即可。需要注意的是，图片的base64编码是不包含图片头的，如data:image/jpg;base64,
* **图片格式**：现支持PNG、JPG、JPEG、BMP，**不支持GIF图片**
* **人脸识别接口分为V2和V3两个版本，本文档为V3版本接口的说明文档，请确认您在百度云后台获得的是V3版本接口权限，再来阅读本文档。**

辨别接口版本的方法是：在百度云后台进入【应用列表】，点击【应用名称】，在【API列表】中可以看到【请求地址】，若请求地址中带有【v3】标识，则您具有的是v3权限，可以阅读本文档；若请求地址中带有【v2】标识，则您具有的是v2权限，应该去阅读v2文档。

**请求示例**

HTTP方法：POST

请求URL： https://aip.baidubce.com/rest/2.0/face/v3/match

URL参数：

| 参数 | 值 |
| --- | --- |
| access\_token | 通过API Key和Secret Key获取的access\_token,参考“[Access Token获取](http://ai.baidu.com/docs#Beginner-Auth)” |

Header：

| 参数 | 值 |
| --- | --- |
| Content-Type | application/json |

Body中放置请求参数，参数详情如下：

**请求参数**

| 参数 | 必选 | 类型 | 说明 |
| --- | --- | --- | --- |
| image | 是 | string | 图片信息(**总数据大小应小于10M**)，图片上传方式根据image\_type来判断。 **两张图片通过json格式上传，格式参考表格下方示例** |
| image\_type | 是 | string | 图片类型 **BASE64**:图片的base64值，base64编码后的图片数据，编码后的图片大小不超过2M； **URL**:图片的 URL地址( 可能由于网络等原因导致下载图片时间过长)； **FACE\_TOKEN**: 人脸图片的唯一标识，调用人脸检测接口时，会为每个人脸图片赋予一个唯一的FACE\_TOKEN，同一张图片多次检测得到的FACE\_TOKEN是同一个。 |
| face\_type | 否 | string | 人脸的类型 **LIVE**：表示生活照：通常为手机、相机拍摄的人像图片、或从网络获取的人像图片等， **IDCARD**：表示身份证芯片照：二代身份证内置芯片中的人像照片， **WATERMARK**：表示带水印证件照：一般为带水印的小图，如公安网小图 **CERT**：表示证件照片：如拍摄的身份证、工卡、护照、学生证等证件图片 默认LIVE |
| quality\_control | 否 | string | 图片质量控制 **NONE**: 不进行控制 **LOW**:较低的质量要求 **NORMAL**: 一般的质量要求 **HIGH**: 较高的质量要求 **默认 NONE** 若图片质量不满足要求，则返回结果中会提示质量检测失败 |
| liveness\_control | 否 | string | 活体检测控制 **NONE**: 不进行控制 **LOW**:较低的活体要求(高通过率 低攻击拒绝率) **NORMAL**: 一般的活体要求(平衡的攻击拒绝率, 通过率) **HIGH**: 较高的活体要求(高攻击拒绝率 低通过率) **默认 NONE** 若活体检测结果不满足要求，则返回结果中会提示活体检测失败 |

说明：两张图片的上传使用json格式，如下：

[

{

"image": "sfasq35sadvsvqwr5q...",

"image\_type": "BASE64",

"face\_type": "LIVE",

"quality\_control": "LOW",

"liveness\_control": "HIGH"

},

{

"image": "sfasq35sadvsvqwr5q...",

"image\_type": "BASE64",

"face\_type": "IDCARD",

"quality\_control": "LOW",

"liveness\_control": "HIGH"

}

]

**请求代码示例**

**提示一**：使用示例代码前，请记得替换其中的示例Token、图片地址或Base64信息。

**提示二**：部分语言依赖的类或库，请在代码注释中查看下载地址。

Java

import com.baidu.ai.aip.utils.HttpUtil;

import com.baidu.ai.aip.utils.GsonUtils;

import java.util.\*;

/\*\*

\* 人脸对比

\*/

public class FaceMatch {

/\*\*

\* 重要提示代码中所需工具类

\* FileUtil,Base64Util,HttpUtil,GsonUtils请从

\* https://ai.baidu.com/file/658A35ABAB2D404FBF903F64D47C1F72

\* https://ai.baidu.com/file/C8D81F3301E24D2892968F09AE1AD6E2

\* https://ai.baidu.com/file/544D677F5D4E4F17B4122FBD60DB82B3

\* https://ai.baidu.com/file/470B3ACCA3FE43788B5A963BF0B625F3

\* 下载

\*/

public static String match() {

// 请求url

String url = "https://aip.baidubce.com/rest/2.0/face/v3/match";

try {

byte[] bytes1 = FileUtil.readFileByBytes("【本地文件1地址】");

byte[] bytes2 = FileUtil.readFileByBytes("【本地文件2地址】");

String image1 = Base64Util.encode(bytes1);

String image2 = Base64Util.encode(bytes2);

List<Map<String, Object>> images = new ArrayList<>();

Map<String, Object> map1 = new HashMap<>();

map1.put("image", image1);

map1.put("image\_type", "BASE64");

map1.put("face\_type", "LIVE");

map1.put("quality\_control", "LOW");

map1.put("liveness\_control", "NORMAL");

Map<String, Object> map2 = new HashMap<>();

map2.put("image", image2);

map2.put("image\_type", "BASE64");

map2.put("face\_type", "LIVE");

map2.put("quality\_control", "LOW");

map2.put("liveness\_control", "NORMAL");

images.add(map1);

images.add(map2);

String param = GsonUtils.toJson(images);

// 注意这里仅为了简化编码每一次请求都去获取access\_token，线上环境access\_token有过期时间， 客户端可自行缓存，过期后重新获取。

String accessToken = "【调用鉴权接口获取的token】";

String result = HttpUtil.post(url, accessToken, "application/json", param);

System.out.println(result);

return result;

} catch (Exception e) {

e.printStackTrace();

}

return null;

}

public static void main(String[] args) {

FaceMatch.match();

}

}

## 2.4.返回说明

**返回参数**

| 参数名 | 必选 | 类型 | 说明 |
| --- | --- | --- | --- |
| score | 是 | float | 人脸相似度得分 |
| face\_list | 是 | array | 人脸信息列表 |
| +face\_token | 是 | string | 人脸的唯一标志 |

* 返回示例
* {
* "score": 44.3,
* "face\_list": [ //返回的顺序与传入的顺序保持一致
* {
* "face\_token": "fid1"
* },
* {
* "face\_token": "fid2"
* }
* ]
* }
* **质量控制参数说明**

不同的控制度下所对应的质量控制阈值，如果检测出来的质量信息某一项不符合控制阈值的要求，则会返回错误信息。

| 控制度 | left\_eye | right\_eye | nose | mouth | left\_cheek | right\_cheek | chin\_contour | illumination | blurdegree | completeness |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| LOW | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 20 | 0.8 | 0 |
| NORMAL | 0.6 | 0.6 | 0.6 | 0.6 | 0.6 | 0.6 | 0.6 | 40 | 0.6 | 0 |
| HIGH | 0.2 | 0.2 | 0.2 | 0.2 | 0.2 | 0.2 | 0.2 | 100 | 0.2 | 1 |

**活体控制参数说明**

不同的控制度下所对应的活体控制阈值，如果检测出来的活体分数小于控制阈值，则会返回错误信息。

| 控制度 | 阈值 | 说明 |
| --- | --- | --- |
| LOW | 0.05 | 万分之一活体误拒率 |
| NORMAL | 0.3 | 千分之一活体误拒率 |
| HIGH | 0.9 | 百分之一活体误拒率 |

误拒率: 把**真人识别为假人**的概率. 阈值越高，安全性越高, 要求也就越高, 对应的误识率就越高

## 2.5.错误码

请参考[人脸识别错误码](http://ai.baidu.com/docs#/Face-ErrorCode-V3/top)

# 3.人脸搜索

**人脸识别接口分为V2和V3两个版本，本文档为V3版本接口的说明文档，请确认您在百度云后台获得的是V3版本接口权限，再来阅读本文档。**

辨别接口版本的方法是：在百度云后台进入【应用列表】，点击【应用名称】，在【API列表】中可以看到【请求地址】，若请求地址中带有【v3】标识，则您具有的是v3权限，可以阅读本文档；若请求地址中带有【v2】标识，则您具有的是v2权限，应该去阅读v2文档。 如果您对文档内容有任何疑问，可以通过以下几种方式联系我们：

* 在百度云控制台内 [提交工单](http://ticket.bce.baidu.com/#/ticket/create)，咨询问题类型请选择**人工智能服务**；
* 如有需要讨论的疑问，欢迎进入 [AI社区](http://ai.baidu.com/forum/topic/list/165) 与其他开发者们一同交流。

## 3.1.能力介绍

**业务能力**

* **1：N人脸搜索**：也称为1：N识别，在指定人脸集合中，找到最相似的人脸；
* **1：N人脸认证**：基于uid维度的1：N识别，由于uid已经锁定固定数量的人脸，所以检索范围更聚焦；

**1：N人脸识别**与**1：N人脸认证**的差别在于：人脸搜索是在指定人脸集合中进行直接地人脸检索操作，而人脸认证是基于uid，先调取这个uid对应的人脸，再在这个uid对应的人脸集合中进行检索（因为每个uid通常对应的只有一张人脸，所以通常也就变为了1：1对比）；实际应用中，人脸认证需要用户或系统先输入id，这增加了验证安全度，但也增加了复杂度，具体使用哪个接口需要视您的业务场景判断。

提示：进行人脸查找相关操作前，建议先阅读 [人脸库管理](http://ai.baidu.com/docs#/Face-Set/dd9c59b9) 相关内容。

## 3.2.调用方式

**请求URL数据格式**

向API服务地址使用POST发送请求，必须在URL中带上参数access\_token，可通过后台的API Key和Secret Key生成，具体方式请参考“[Access Token获取](http://ai.baidu.com/docs#/Auth)”。

**获取access\_token的示例代码**

Java

package com.baidu.ai.aip.auth;

import org.json.JSONObject;

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.net.HttpURLConnection;

import java.net.URL;

import java.util.List;

import java.util.Map;

/\*\*

\* 获取token类

\*/

public class AuthService {

/\*\*

\* 获取权限token

\* @return 返回示例：

\* {

\* "access\_token": "24.460da4889caad24cccdb1fea17221975.2592000.1491995545.282335-1234567",

\* "expires\_in": 2592000

\* }

\*/

public static String getAuth() {

// 官网获取的 API Key 更新为你注册的

String clientId = "百度云应用的AK";

// 官网获取的 Secret Key 更新为你注册的

String clientSecret = "百度云应用的SK";

return getAuth(clientId, clientSecret);

}

/\*\*

\* 获取API访问token

\* 该token有一定的有效期，需要自行管理，当失效时需重新获取.

\* @param ak - 百度云官网获取的 API Key

\* @param sk - 百度云官网获取的 Securet Key

\* @return assess\_token 示例：

\* "24.460da4889caad24cccdb1fea17221975.2592000.1491995545.282335-1234567"

\*/

public static String getAuth(String ak, String sk) {

// 获取token地址

String authHost = "https://aip.baidubce.com/oauth/2.0/token?";

String getAccessTokenUrl = authHost

// 1. grant\_type为固定参数

+ "grant\_type=client\_credentials"

// 2. 官网获取的 API Key

+ "&client\_id=" + ak

// 3. 官网获取的 Secret Key

+ "&client\_secret=" + sk;

try {

URL realUrl = new URL(getAccessTokenUrl);

// 打开和URL之间的连接

HttpURLConnection connection = (HttpURLConnection) realUrl.openConnection();

connection.setRequestMethod("GET");

connection.connect();

// 获取所有响应头字段

Map<String, List<String>> map = connection.getHeaderFields();

// 遍历所有的响应头字段

for (String key : map.keySet()) {

System.err.println(key + "--->" + map.get(key));

}

// 定义 BufferedReader输入流来读取URL的响应

BufferedReader in = new BufferedReader(new InputStreamReader(connection.getInputStream()));

String result = "";

String line;

while ((line = in.readLine()) != null) {

result += line;

}

/\*\*

\* 返回结果示例

\*/

System.err.println("result:" + result);

JSONObject jsonObject = new JSONObject(result);

String access\_token = jsonObject.getString("access\_token");

return access\_token;

} catch (Exception e) {

System.err.printf("获取token失败！");

e.printStackTrace(System.err);

}

return null;

}

}

**注意**：access\_token的有效期为30天，**切记需要每30天进行定期更换，或者每次请求都拉取新token**；

例如此接口，使用HTTPS POST发送：

https://aip.baidubce.com/rest/2.0/face/v3/search?access\_token=24.f9ba9c5341b67688ab4added8bc91dec.2592000.1485570332.282335-8574074

POST中Body的参数，按照下方请求参数说明选择即可。

**提示**：如果您为百度云老用户，正在使用其他非AI的服务，可以参考[百度云AKSK鉴权方式](https://cloud.baidu.com/doc/Reference/AuthenticationMechanism.html)发送请求，虽然**请求方式**和**鉴权方法**和本文所介绍的不同，但请求参数和返回结果一致。

## 3.3.请求说明

**注意事项**：

* **请求体格式化**：Content-Type为application/json，通过json格式化请求体。
* **Base64编码**：请求的图片需经过Base64编码，图片的base64编码指将图片数据编码成一串字符串，使用该字符串代替图像地址。您可以首先得到图片的二进制，然后用Base64格式编码即可。需要注意的是，图片的base64编码是不包含图片头的，如data:image/jpg;base64,
* **图片格式**：现支持PNG、JPG、JPEG、BMP，**不支持GIF图片**
* **人脸识别接口分为V2和V3两个版本，本文档为V3版本接口的说明文档，请确认您在百度云后台获得的是V3版本接口权限，再来阅读本文档。**

辨别接口版本的方法是：在百度云后台进入【应用列表】，点击【应用名称】，在【API列表】中可以看到【请求地址】，若请求地址中带有【v3】标识，则您具有的是v3权限，可以阅读本文档；若请求地址中带有【v2】标识，则您具有的是v2权限，应该去阅读v2文档。

**请求示例**

HTTP方法：POST

请求URL： https://aip.baidubce.com/rest/2.0/face/v3/search

URL参数：

| 参数 | 值 |
| --- | --- |
| access\_token | 通过API Key和Secret Key获取的access\_token,参考“[Access Token获取](http://ai.baidu.com/docs#/Auth)” |

Header如下：

| 参数 | 值 |
| --- | --- |
| Content-Type | application/json |

Body中放置请求参数，参数详情如下：

**请求参数**

| 参数 | 必选 | 类型 | 说明 |
| --- | --- | --- | --- |
| image | 是 | string | 图片信息(**总数据大小应小于10M**)，图片上传方式根据image\_type来判断 |
| image\_type | 是 | string | 图片类型 **BASE64**:图片的base64值，base64编码后的图片数据，编码后的图片大小不超过2M； **URL**:图片的 URL地址( 可能由于网络等原因导致下载图片时间过长)； **FACE\_TOKEN**: 人脸图片的唯一标识，调用人脸检测接口时，会为每个人脸图片赋予一个唯一的FACE\_TOKEN，同一张图片多次检测得到的FACE\_TOKEN是同一个。 |
| group\_id\_list | 是 | string | 从指定的group中进行查找 用逗号分隔，**上限20个** |
| quality\_control | 否 | string | 图片质量控制 **NONE**: 不进行控制 **LOW**:较低的质量要求 **NORMAL**: 一般的质量要求 **HIGH**: 较高的质量要求 **默认 NONE** 若图片质量不满足要求，则返回结果中会提示质量检测失败 |
| liveness\_control | 否 | string | 活体检测控制 **NONE**: 不进行控制 **LOW**:较低的活体要求(高通过率 低攻击拒绝率) **NORMAL**: 一般的活体要求(平衡的攻击拒绝率, 通过率) **HIGH**: 较高的活体要求(高攻击拒绝率 低通过率) **默认NONE** 若活体检测结果不满足要求，则返回结果中会提示活体检测失败 |
| user\_id | 否 | string | 当需要对特定用户进行比对时，指定user\_id进行比对。即人脸认证功能。 |
| max\_user\_num | 否 | unit32 | 查找后返回的用户数量。返回相似度最高的几个用户，默认为1，最多返回20个。 |

说明：如果使用base 64格式的图片，两张请求的图片请分别进行base64编码。

**请求代码示例**

**提示一**：使用示例代码前，请记得替换其中的示例Token、图片地址或Base64信息。

**提示二**：部分语言依赖的类或库，请在代码注释中查看下载地址。

Java

import com.baidu.ai.aip.utils.HttpUtil;

import com.baidu.ai.aip.utils.GsonUtils;

import java.util.\*;

/\*\*

\* 人脸搜索

\*/

public class FaceSearch {

/\*\*

\* 重要提示代码中所需工具类

\* FileUtil,Base64Util,HttpUtil,GsonUtils请从

\* https://ai.baidu.com/file/658A35ABAB2D404FBF903F64D47C1F72

\* https://ai.baidu.com/file/C8D81F3301E24D2892968F09AE1AD6E2

\* https://ai.baidu.com/file/544D677F5D4E4F17B4122FBD60DB82B3

\* https://ai.baidu.com/file/470B3ACCA3FE43788B5A963BF0B625F3

\* 下载

\*/

public static String search() {

// 请求url

String url = "https://aip.baidubce.com/rest/2.0/face/v3/search";

try {

Map<String, Object> map = new HashMap<>();

map.put("image", "027d8308a2ec665acb1bdf63e513bcb9");

map.put("liveness\_control", "NORMAL");

map.put("group\_id\_list", "group\_repeat,group\_233");

map.put("image\_type", "FACE\_TOKEN");

map.put("quality\_control", "LOW");

String param = GsonUtils.toJson(map);

// 注意这里仅为了简化编码每一次请求都去获取access\_token，线上环境access\_token有过期时间， 客户端可自行缓存，过期后重新获取。

String accessToken = "[调用鉴权接口获取的token]";

String result = HttpUtil.post(url, accessToken, "application/json", param);

System.out.println(result);

return result;

} catch (Exception e) {

e.printStackTrace();

}

return null;

}

public static void main(String[] args) {

FaceSearch.search();

}

}

## 3.4.返回说明

**返回参数**

* 返回结果

| 字段 | 必选 | 类型 | 说明 |
| --- | --- | --- | --- |
| face\_token | 是 | string | 人脸标志 |
| user\_list | 是 | array | 匹配的用户信息列表 |
| +group\_id | 是 | string | 用户所属的group\_id |
| +user\_id | 是 | string | 用户的user\_id |
| +user\_info | 是 | string | 注册用户时携带的user\_info |
| +score | 是 | float | 用户的匹配得分 |

* 返回示例

{

"face\_token": "fid",

"user\_list": [

{

"group\_id" : "test1",

"user\_id": "u333333",

"user\_info": "Test User",

"score": 99.3

}

]

}

* **质量控制参数说明**

不同的控制度下所对应的质量控制阈值，如果检测出来的质量信息某一项不符合控制阈值的要求，则会返回错误信息。

| 控制度 | left\_eye | right\_eye | nose | mouth | left\_cheek | right\_cheek | chin\_contour | illumination | blurdegree | completeness |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| LOW | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 20 | 0.8 | 0 |
| NORMAL | 0.6 | 0.6 | 0.6 | 0.6 | 0.6 | 0.6 | 0.6 | 40 | 0.6 | 0 |
| HIGH | 0.2 | 0.2 | 0.2 | 0.2 | 0.2 | 0.2 | 0.2 | 100 | 0.2 | 1 |

**活体控制参数说明**

不同的控制度下所对应的活体控制阈值，如果检测出来的活体分数小于控制阈值，则会返回错误信息。

| 控制度 | 阈值 | 说明 |
| --- | --- | --- |
| LOW | 0.05 | 万分之一活体误拒率 |
| NORMAL | 0.3 | 千分之一活体误拒率 |
| HIGH | 0.9 | 百分之一活体误拒率 |

误拒率: 把**真人识别为假人**的概率. 阈值越高，安全性越高, 要求也就越高, 对应的误识率就越高

## 3.5.错误码

请参考[人脸识别错误码](http://ai.baidu.com/docs#/Face-ErrorCode-V3/top)

# 4.人脸库管理

**人脸识别接口分为V2和V3两个版本，本文档为V3版本接口的说明文档，请确认您在百度云后台获得的是V3版本接口权限，再来阅读本文档。**

辨别接口版本的方法是：在百度云后台进入【应用列表】，点击【应用名称】，在【API列表】中可以看到【请求地址】，若请求地址中带有【v3】标识，则您具有的是v3权限，可以阅读本文档；若请求地址中带有【v2】标识，则您具有的是v2权限，应该去阅读v2文档。 如果您对文档内容有任何疑问，可以通过以下几种方式联系我们：

* 在百度云控制台内 [提交工单](http://ticket.bce.baidu.com/#/ticket/create)，咨询问题类型请选择**人工智能服务**；
* 如有需要讨论的疑问，欢迎进入 [AI社区](http://ai.baidu.com/forum/topic/list/165) 与其他开发者们一同交流。

## 4.1.能力介绍

**业务能力**

人脸库管理相关接口，要完成1：N或者M：N识别，首先需要构建一个人脸库，用于存放所有人脸特征，相关接口如下：

* **人脸注册**：向人脸库中添加人脸
* **人脸更新**：更新人脸库中指定用户下的人脸信息
* **人脸删除**：删除指定用户的某张人脸
* **用户信息查询**：查询人脸库中某个用户的详细信息
* **获取用户人脸列表**：获取某个用户组中的全部人脸列表
* **获取用户列表**：查询指定用户组中的用户列表
* **复制用户**：将指定用户复制到另外的人脸组
* **删除用户**：删除指定用户
* **创建用户组**：创建一个新的用户组
* **删除用户组**：删除指定用户组
* **组列表查询**：查询人脸库中用户组的列表

**人脸库结构**

人脸库、用户组、用户、用户下的人脸**层级关系**如下所示：

|- 人脸库(appid)

|- 用户组一（group\_id）

|- 用户01（uid）

|- 人脸（faceid）

|- 用户02（uid）

|- 人脸（faceid）

|- 人脸（faceid）

....

....

|- 用户组二（group\_id）

|- 用户组三（group\_id）

....

**关于人脸库的设置限制**

* 每个开发者账号可以创建100个appid；
* **每个appid对应一个人脸库，且不同appid之间，人脸库互不相通**；
* 每个人脸库下，可以创建多个用户组，用户组（group）数量**没有限制**；
* 每个用户组（group）下，可添加**无限**个user\_id，**无限**张人脸（注：为了保证查询速度，单个group中的人脸容量上限建议为**80万**）；
* 每个用户（user\_id）所能注册的最大人脸数量**没有限制**；

提醒：每个人脸库对应一个appid，一定确保不要轻易删除后台应用列表中的appid，删除后则此人脸库将失效，无法进行任何查找！

**质量判断**

为了保证识别效果，请控制注册人脸的质量，在调用人脸注册接口时使用质量控制和活体控制参数保证图片的质量以及注册进入人脸库的人脸是活体：

## 4.2.调用方式

**请求URL数据格式**

向API服务地址使用POST发送请求，必须在URL中带上参数access\_token，可通过后台的API Key和Secret Key生成，具体方式请参考“[Access Token获取](http://ai.baidu.com/docs#/Auth)”。

**获取access\_token的示例代码**

Java

package com.baidu.ai.aip.auth;

import org.json.JSONObject;

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.net.HttpURLConnection;

import java.net.URL;

import java.util.List;

import java.util.Map;

/\*\*

\* 获取token类

\*/

public class AuthService {

/\*\*

\* 获取权限token

\* @return 返回示例：

\* {

\* "access\_token": "24.460da4889caad24cccdb1fea17221975.2592000.1491995545.282335-1234567",

\* "expires\_in": 2592000

\* }

\*/

public static String getAuth() {

// 官网获取的 API Key 更新为你注册的

String clientId = "百度云应用的AK";

// 官网获取的 Secret Key 更新为你注册的

String clientSecret = "百度云应用的SK";

return getAuth(clientId, clientSecret);

}

/\*\*

\* 获取API访问token

\* 该token有一定的有效期，需要自行管理，当失效时需重新获取.

\* @param ak - 百度云官网获取的 API Key

\* @param sk - 百度云官网获取的 Securet Key

\* @return assess\_token 示例：

\* "24.460da4889caad24cccdb1fea17221975.2592000.1491995545.282335-1234567"

\*/

public static String getAuth(String ak, String sk) {

// 获取token地址

String authHost = "https://aip.baidubce.com/oauth/2.0/token?";

String getAccessTokenUrl = authHost

// 1. grant\_type为固定参数

+ "grant\_type=client\_credentials"

// 2. 官网获取的 API Key

+ "&client\_id=" + ak

// 3. 官网获取的 Secret Key

+ "&client\_secret=" + sk;

try {

URL realUrl = new URL(getAccessTokenUrl);

// 打开和URL之间的连接

HttpURLConnection connection = (HttpURLConnection) realUrl.openConnection();

connection.setRequestMethod("GET");

connection.connect();

// 获取所有响应头字段

Map<String, List<String>> map = connection.getHeaderFields();

// 遍历所有的响应头字段

for (String key : map.keySet()) {

System.err.println(key + "--->" + map.get(key));

}

// 定义 BufferedReader输入流来读取URL的响应

BufferedReader in = new BufferedReader(new InputStreamReader(connection.getInputStream()));

String result = "";

String line;

while ((line = in.readLine()) != null) {

result += line;

}

/\*\*

\* 返回结果示例

\*/

System.err.println("result:" + result);

JSONObject jsonObject = new JSONObject(result);

String access\_token = jsonObject.getString("access\_token");

return access\_token;

} catch (Exception e) {

System.err.printf("获取token失败！");

e.printStackTrace(System.err);

}

return null;

}

}

**注意**：access\_token的有效期为30天，**切记需要每30天进行定期更换，或者每次请求都拉取新token**；

例如此接口，使用HTTPS POST发送：

https://aip.baidubce.com/rest/2.0/face/v3/faceset/user/add?access\_token=24.f9ba9c5341b67688ab6added8bc91dec.2592000.1485570332.282335-8574074

POST中Body的参数，按照下方请求参数说明选择即可。

**提示**：如果您为百度云老用户，正在使用其他非AI的服务，可以参考[百度云AKSK鉴权方式](https://cloud.baidu.com/doc/Reference/AuthenticationMechanism.html)发送请求，虽然**请求方式**和**鉴权方法**和本文所介绍的不同，但请求参数和返回结果一致。

# 5.人脸注册

## 5.1.接口描述

用于向人脸库中新增用户，及组内用户的人脸图片，

典型应用场景：构建您的人脸库，如**会员人脸注册**，**已有用户补全人脸信息**等。

## 5.2.请求说明

**注意事项**：

* **请求体格式化**：Content-Type为application/json，通过json格式化请求体。
* **Base64编码**：请求的图片需经过Base64编码，图片的base64编码指将图片数据编码成一串字符串，使用该字符串代替图像地址。您可以首先得到图片的二进制，然后用Base64格式编码即可。需要注意的是，图片的base64编码是不包含图片头的，如data:image/jpg;base64,
* **图片格式**：现支持PNG、JPG、JPEG、BMP，**不支持GIF图片**
* **人脸识别接口分为V2和V3两个版本，本文档为V3版本接口的说明文档，请确认您在百度云后台获得的是V3版本接口权限，再来阅读本文档。**

辨别接口版本的方法是：在百度云后台进入【应用列表】，点击【应用名称】，在【API列表】中可以看到【请求地址】，若请求地址中带有【v3】标识，则您具有的是v3权限，可以阅读本文档；若请求地址中带有【v2】标识，则您具有的是v2权限，应该去阅读v2文档。

**请求示例**

HTTP方法：POST

请求URL： https://aip.baidubce.com/rest/2.0/face/v3/faceset/user/add

URL参数：

| 参数 | 值 |
| --- | --- |
| access\_token | 通过API Key和Secret Key获取的access\_token,参考“[Access Token获取](http://ai.baidu.com/docs#/Auth)” |

Header如下：

| 参数 | 值 |
| --- | --- |
| Content-Type | application/json |

Body中放置请求参数，参数详情如下：

**请求参数**

| 参数 | 必选 | 类型 | 说明 |
| --- | --- | --- | --- |
| image | 是 | string | 图片信息(**总数据大小应小于10M**)，图片上传方式根据image\_type来判断 |
| image\_type | 是 | string | 图片类型 **BASE64**:图片的base64值，base64编码后的图片数据，编码后的图片大小不超过2M； **URL**:图片的 URL地址( 可能由于网络等原因导致下载图片时间过长)； **FACE\_TOKEN**：人脸图片的唯一标识，调用人脸检测接口时，会为每个人脸图片赋予一个唯一的FACE\_TOKEN，同一张图片多次检测得到的FACE\_TOKEN是同一个。 |
| group\_id | 是 | string | 用户组id，标识一组用户（由数字、字母、下划线组成），长度限制128B。**产品建议**：根据您的业务需求，可以将需要注册的用户，按照业务划分，分配到不同的group下，例如按照会员手机尾号作为groupid，用于刷脸支付、会员计费消费等，这样可以尽可能控制每个group下的用户数与人脸数，提升检索的准确率 |
| user\_id | 是 | string | 用户id（由数字、字母、下划线组成），长度限制128B |
| user\_info | 否 | string | 用户资料，长度限制256B 默认空 |
| quality\_control | 否 | string | 图片质量控制 **NONE**: 不进行控制 **LOW**:较低的质量要求 **NORMAL**: 一般的质量要求 **HIGH**: 较高的质量要求 **默认 NONE** 若图片质量不满足要求，则返回结果中会提示质量检测失败 |
| liveness\_control | 否 | string | 活体检测控制 **NONE**: 不进行控制 **LOW**:较低的活体要求(高通过率 低攻击拒绝率) **NORMAL**: 一般的活体要求(平衡的攻击拒绝率, 通过率) **HIGH**: 较高的活体要求(高攻击拒绝率 低通过率) **默认NONE** 若活体检测结果不满足要求，则返回结果中会提示活体检测失败 |

**说明**：人脸注册完毕后，生效时间**一般为5s以内**，之后便可以进行人脸搜索或认证操作。

**请求代码示例**

**提示一**：使用示例代码前，请记得替换其中的示例Token、图片地址或Base64信息。

**提示二**：部分语言依赖的类或库，请在代码注释中查看下载地址。

Java

import com.baidu.ai.aip.utils.HttpUtil;

import com.baidu.ai.aip.utils.GsonUtils;

import java.util.\*;

/\*\*

\* 人脸注册

\*/

public class FaceAdd {

/\*\*

\* 重要提示代码中所需工具类

\* FileUtil,Base64Util,HttpUtil,GsonUtils请从

\* https://ai.baidu.com/file/658A35ABAB2D404FBF903F64D47C1F72

\* https://ai.baidu.com/file/C8D81F3301E24D2892968F09AE1AD6E2

\* https://ai.baidu.com/file/544D677F5D4E4F17B4122FBD60DB82B3

\* https://ai.baidu.com/file/470B3ACCA3FE43788B5A963BF0B625F3

\* 下载

\*/

public static String add() {

// 请求url

String url = "https://aip.baidubce.com/rest/2.0/face/v3/faceset/user/add";

try {

Map<String, Object> map = new HashMap<>();

map.put("image", "027d8308a2ec665acb1bdf63e513bcb9");

map.put("group\_id", "group\_repeat");

map.put("user\_id", "user1");

map.put("user\_info", "abc");

map.put("liveness\_control", "NORMAL");

map.put("image\_type", "FACE\_TOKEN");

map.put("quality\_control", "LOW");

String param = GsonUtils.toJson(map);

// 注意这里仅为了简化编码每一次请求都去获取access\_token，线上环境access\_token有过期时间， 客户端可自行缓存，过期后重新获取。

String accessToken = "[调用鉴权接口获取的token]";

String result = HttpUtil.post(url, accessToken, "application/json", param);

System.out.println(result);

return result;

} catch (Exception e) {

e.printStackTrace();

}

return null;

}

public static void main(String[] args) {

FaceAdd.add();

}

}

## 5.3.返回说明

**返回参数**

| 字段 | 必选 | 类型 | 说明 |
| --- | --- | --- | --- |
| log\_id | 是 | uint64 | 请求标识码，随机数，唯一 |
| face\_token | 是 | string | 人脸图片的唯一标识 |
| location | 是 | array | 人脸在图片中的位置 |
| +left | 是 | double | 人脸区域离左边界的距离 |
| +top | 是 | double | 人脸区域离上边界的距离 |
| +width | 是 | double | 人脸区域的宽度 |
| +height | 是 | double | 人脸区域的高度 |
| +rotation | 是 | int64 | 人脸框相对于竖直方向的顺时针旋转角，[-180,180] |

**返回示例**

{

"face\_token": "2fa64a88a9d5118916f9a303782a97d3",

"location": {

"left": 117,

"top": 131,

"width": 172,

"height": 170,

"rotation": 4

}

}

* **质量控制参数说明**

不同的控制度下所对应的质量控制阈值，如果检测出来的质量信息某一项不符合控制阈值的要求，则会返回错误信息。

| 控制度 | left\_eye | right\_eye | nose | mouth | left\_cheek | right\_cheek | chin\_contour | illumination | blurdegree | completeness |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| LOW | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 20 | 0.8 | 0 |
| NORMAL | 0.6 | 0.6 | 0.6 | 0.6 | 0.6 | 0.6 | 0.6 | 40 | 0.6 | 0 |
| HIGH | 0.2 | 0.2 | 0.2 | 0.2 | 0.2 | 0.2 | 0.2 | 100 | 0.2 | 1 |

**活体控制参数说明**

不同的控制度下所对应的活体控制阈值，如果检测出来的活体分数小于控制阈值，则会返回错误信息。

| 控制度 | 阈值 | 说明 |
| --- | --- | --- |
| LOW | 0.05 | 万分之一活体误拒率 |
| NORMAL | 0.3 | 千分之一活体误拒率 |
| HIGH | 0.9 | 百分之一活体误拒率 |

误拒率: 把**真人识别为假人**的概率. 阈值越高，安全性越高, 要求也就越高, 对应的误识率就越高

# 6.人脸更新

## 6.1.接口描述

用于对人脸库中指定用户，更新其下的人脸图像。

**说明**：针对一个user\_id执行更新操作，新上传的人脸图像将覆盖此user\_id原有所有图像。

## 6.2.请求说明

**注意事项**：

* **请求体格式化**：Content-Type为application/json，通过json格式化请求体。
* **Base64编码**：请求的图片需经过Base64编码，图片的base64编码指将图片数据编码成一串字符串，使用该字符串代替图像地址。您可以首先得到图片的二进制，然后用Base64格式编码即可。需要注意的是，图片的base64编码是不包含图片头的，如data:image/jpg;base64,
* **图片格式**：现支持PNG、JPG、JPEG、BMP，**不支持GIF图片**
* **人脸识别接口分为V2和V3两个版本，本文档为V3版本接口的说明文档，请确认您在百度云后台获得的是V3版本接口权限，再来阅读本文档。**

辨别接口版本的方法是：在百度云后台进入【应用列表】，点击【应用名称】，在【API列表】中可以看到【请求地址】，若请求地址中带有【v3】标识，则您具有的是v3权限，可以阅读本文档；若请求地址中带有【v2】标识，则您具有的是v2权限，应该去阅读v2文档。

**请求示例**

HTTP方法：POST

请求URL： https://aip.baidubce.com/rest/2.0/face/v3/faceset/user/update

URL参数：

| 参数 | 值 |
| --- | --- |
| access\_token | 通过API Key和Secret Key获取的access\_token,参考“[Access Token获取](http://ai.baidu.com/docs#/Auth)” |

Header如下：

| 参数 | 值 |
| --- | --- |
| Content-Type | application/json |

Body中放置请求参数，参数详情如下：

**请求参数**

| 参数 | 必选 | 类型 | 说明 |
| --- | --- | --- | --- |
| image | 是 | string | 图片信息(**总数据大小应小于10M**)，图片上传方式根据image\_type来判断 |
| image\_type | 是 | string | 图片类型 **BASE64**:图片的base64值，base64编码后的图片数据，编码后的图片大小不超过2M； **URL**:图片的 URL地址( 可能由于网络等原因导致下载图片时间过长)； **FACE\_TOKEN**: 人脸图片的唯一标识 |
| group\_id | 是 | string | 用户组id，标识一组用户（由数字、字母、下划线组成），长度限制128B |
| user\_id | 是 | string | 用户id（由数字、字母、下划线组成），长度限制128B |
| user\_info | 否 | string | 用户资料，长度限制256B 默认空 |
| quality\_control | 否 | string | 图片质量控制 **NONE**: 不进行控制 **LOW**:较低的质量要求 **NORMAL**: 一般的质量要求 **HIGH**: 较高的质量要求 **默认 NONE** 若图片质量不满足要求，则返回结果中会提示质量检测失败 |
| liveness\_control | 否 | string | 活体检测控制 **NONE**: 不进行控制 **LOW**:较低的活体要求(高通过率 低攻击拒绝率) **NORMAL**: 一般的活体要求(平衡的攻击拒绝率, 通过率) **HIGH**: 较高的活体要求(高攻击拒绝率 低通过率) **默认NONE** 若活体检测结果不满足要求，则返回结果中会提示活体检测失败 |

**请求代码示例**

**提示一**：使用示例代码前，请记得替换其中的示例Token、图片地址或Base64信息。

**提示二**：部分语言依赖的类或库，请在代码注释中查看下载地址。

Java

import com.baidu.ai.aip.utils.HttpUtil;

import com.baidu.ai.aip.utils.GsonUtils;

import java.util.\*;

/\*\*

\* 人脸更新

\*/

public class FaceUpdate {

/\*\*

\* 重要提示代码中所需工具类

\* FileUtil,Base64Util,HttpUtil,GsonUtils请从

\* https://ai.baidu.com/file/658A35ABAB2D404FBF903F64D47C1F72

\* https://ai.baidu.com/file/C8D81F3301E24D2892968F09AE1AD6E2

\* https://ai.baidu.com/file/544D677F5D4E4F17B4122FBD60DB82B3

\* https://ai.baidu.com/file/470B3ACCA3FE43788B5A963BF0B625F3

\* 下载

\*/

public static String update() {

// 请求url

String url = "https://aip.baidubce.com/rest/2.0/face/v3/faceset/user/update";

try {

Map<String, Object> map = new HashMap<>();

map.put("image", "027d8308a2ec665acb1bdf63e513bcb9");

map.put("group\_id", "group\_repeat");

map.put("user\_id", "user1");

map.put("user\_info", "cba");

map.put("liveness\_control", "NORMAL");

map.put("image\_type", "FACE\_TOKEN");

map.put("quality\_control", "LOW");

String param = GsonUtils.toJson(map);

// 注意这里仅为了简化编码每一次请求都去获取access\_token，线上环境access\_token有过期时间， 客户端可自行缓存，过期后重新获取。

String accessToken = "[调用鉴权接口获取的token]";

String result = HttpUtil.post(url, accessToken, "application/json", param);

System.out.println(result);

return result;

} catch (Exception e) {

e.printStackTrace();

}

return null;

}

public static void main(String[] args) {

FaceUpdate.update();

}

}

## 6.3.返回说明

**返回参数**

| 字段 | 必选 | 类型 | 说明 |
| --- | --- | --- | --- |
| log\_id | 是 | uint64 | 请求标识码，随机数，唯一 |
| face\_token | 是 | string | 人脸图片的唯一标识 |
| location | 是 | array | 人脸在图片中的位置 |
| +left | 是 | double | 人脸区域离左边界的距离 |
| +top | 是 | double | 人脸区域离上边界的距离 |
| +width | 是 | double | 人脸区域的宽度 |
| +height | 是 | double | 人脸区域的高度 |
| +rotation | 是 | int64 | 人脸框相对于竖直方向的顺时针旋转角，[-180,180] |

**返回示例**

{

"face\_token": "2fa64a88a9d5118916f9a303782a97d3",

"location": {

"left": 117,

"top": 131,

"width": 172,

"height": 170,

"rotation": 4

}

}

# 7.人脸删除

## 7.1.接口描述

删除用户的某一张人脸，如果该用户只有一张人脸图片，则同时删除用户。

## 7.2.请求说明

**请求示例**

HTTP方法：POST

请求URL： https://aip.baidubce.com/rest/2.0/face/v3/faceset/face/delete

URL参数：

| 参数 | 值 |
| --- | --- |
| access\_token | 通过API Key和Secret Key获取的access\_token,参考“[Access Token获取](http://ai.baidu.com/docs#/Auth)” |

Header如下：

| 参数 | 值 |
| --- | --- |
| Content-Type | application/json |

Body中放置请求参数，参数详情如下：

**请求参数**

| 参数 | 必选 | 类型 | 说明 |
| --- | --- | --- | --- |
| user\_id | 是 | string | 用户id（由数字、字母、下划线组成），长度限制128B |
| group\_id | 是 | string | 用户组id（由数字、字母、下划线组成） 长度限制48B，删除指定group\_id中的user\_id信息 |
| face\_token | 是 | string | 需要删除的人脸图片token，（由数字、字母、下划线组成）长度限制64B |

**请求代码示例**

**提示一**：使用示例代码前，请记得替换其中的示例Token、图片地址或Base64信息。

**提示二**：部分语言依赖的类或库，请在代码注释中查看下载地址。

undefined

## 7.3.返回说明

通过返回的error\_code判断是否成功 如失败则查看error\_msg获得具体错误信息

**返回示例**

// 删除成功

{

"error\_code": 0,

"log\_id": 73473737,

}

// 删除发生错误

{

"error\_code": 223106,

"log\_id": 1382953199,

"error\_msg": "face is not exist"

}

# 8.用户信息查询

## 8.1.接口描述

获取人脸库中某个用户的信息(user\_info信息和用户所属的组)。

## 8.2.请求说明

**请求示例**

HTTP方法：POST

请求URL： https://aip.baidubce.com/rest/2.0/face/v3/faceset/user/get

URL参数：

| 参数 | 值 |
| --- | --- |
| access\_token | 通过API Key和Secret Key获取的access\_token,参考“[Access Token获取](http://ai.baidu.com/docs#/Auth)” |

Header如下：

| 参数 | 值 |
| --- | --- |
| Content-Type | application/json |

Body中放置请求参数，参数详情如下：

**请求参数**

| 参数 | 必选 | 类型 | 说明 |
| --- | --- | --- | --- |
| user\_id | 是 | string | 用户id（由数字、字母、下划线组成），长度限制128B |
| group\_id | 是 | string | 用户组id(由数字、字母、下划线组成，长度限制48B)，如传入“**@ALL**”则从所有组中查询用户信息。注：处于不同组，但uid相同的用户，我们认为是同一个用户。 |

**请求代码示例**

**提示一**：使用示例代码前，请记得替换其中的示例Token、图片地址或Base64信息。

**提示二**：部分语言依赖的类或库，请在代码注释中查看下载地址。

Java

package com.baidu.ai.aip;

import com.baidu.ai.aip.utils.HttpUtil;

import com.baidu.ai.aip.utils.GsonUtils;

import java.util.\*;

/\*\*

\* 用户信息查询

\*/

public class UserGet {

/\*\*

\* 重要提示代码中所需工具类

\* FileUtil,Base64Util,HttpUtil,GsonUtils请从

\* https://ai.baidu.com/file/658A35ABAB2D404FBF903F64D47C1F72

\* https://ai.baidu.com/file/C8D81F3301E24D2892968F09AE1AD6E2

\* https://ai.baidu.com/file/544D677F5D4E4F17B4122FBD60DB82B3

\* https://ai.baidu.com/file/470B3ACCA3FE43788B5A963BF0B625F3

\* 下载

\*/

public static String get() {

// 请求url

String url = "https://aip.baidubce.com/rest/2.0/face/v3/faceset/user/get";

try {

Map<String, Object> map = new HashMap<>();

map.put("user\_id", "user1");

map.put("group\_id", "group1");

String param = GsonUtils.toJson(map);

// 注意这里仅为了简化编码每一次请求都去获取access\_token，线上环境access\_token有过期时间， 客户端可自行缓存，过期后重新获取。

String accessToken = "[调用鉴权接口获取的token]";

String result = HttpUtil.post(url, accessToken, "application/json", param);

System.out.println(result);

return result;

} catch (Exception e) {

e.printStackTrace();

}

return null;

}

public static void main(String[] args) {

UserGet.get();

}

}

## 8.3.返回说明

**返回参数**

| 字段 | 必选 | 类型 | 说明 |
| --- | --- | --- | --- |
| log\_id | 是 | uint64 | 请求标识码，随机数，唯一 |
| user\_list | 是 | array | 查询到的用户列表 |
| +user\_info | 是 | string | 用户资料，被查询用户的资料 |
| +group\_id | 是 | string | 用户组id，被查询用户的所在组 |

**返回示例**

{

"user\_list": [

{

"user\_info": "user info ...",

"group\_id": "gid1"

},

{

"user\_info": "user info2 ...",

"group\_id": "gid2"

}

]

}

# 9.获取用户人脸列表

## 9.1.接口描述

用于获取一个用户的全部人脸列表。

## 9.2.请求说明

**请求示例**

HTTP方法：POST

请求URL： https://aip.baidubce.com/rest/2.0/face/v3/faceset/face/getlist

URL参数：

| 参数 | 值 |
| --- | --- |
| access\_token | 通过API Key和Secret Key获取的access\_token,参考“[Access Token获取](http://ai.baidu.com/docs#/Auth)” |

Header如下：

| 参数 | 值 |
| --- | --- |
| Content-Type | application/json |

Body中放置请求参数，参数详情如下：

**请求参数**

| 参数 | 必选 | 类型 | 说明 |
| --- | --- | --- | --- |
| user\_id | 是 | string | 用户id（由数字、字母、下划线组成），长度限制128B |
| group\_id | 是 | string | 用户组id (由数字、字母、下划线组成），长度限制48B |

**请求代码示例**

**提示一**：使用示例代码前，请记得替换其中的示例Token、图片地址或Base64信息。

**提示二**：部分语言依赖的类或库，请在代码注释中查看下载地址。

Java

package com.baidu.ai.aip;

import com.baidu.ai.aip.utils.HttpUtil;

import com.baidu.ai.aip.utils.GsonUtils;

import java.util.\*;

/\*\*

\* 获取用户人脸列表

\*/

public class FaceGetList {

/\*\*

\* 重要提示代码中所需工具类

\* FileUtil,Base64Util,HttpUtil,GsonUtils请从

\* https://ai.baidu.com/file/658A35ABAB2D404FBF903F64D47C1F72

\* https://ai.baidu.com/file/C8D81F3301E24D2892968F09AE1AD6E2

\* https://ai.baidu.com/file/544D677F5D4E4F17B4122FBD60DB82B3

\* https://ai.baidu.com/file/470B3ACCA3FE43788B5A963BF0B625F3

\* 下载

\*/

public static String getList() {

// 请求url

String url = "https://aip.baidubce.com/rest/2.0/face/v3/faceset/face/getlist";

try {

Map<String, Object> map = new HashMap<>();

map.put("user\_id", "user1");

map.put("group\_id", "group1");

String param = GsonUtils.toJson(map);

// 注意这里仅为了简化编码每一次请求都去获取access\_token，线上环境access\_token有过期时间， 客户端可自行缓存，过期后重新获取。

String accessToken = "[调用鉴权接口获取的token]";

String result = HttpUtil.post(url, accessToken, "application/json", param);

System.out.println(result);

return result;

} catch (Exception e) {

e.printStackTrace();

}

return null;

}

public static void main(String[] args) {

FaceGetList.getList();

}

}

## 9.3.返回说明

**返回参数**

| 字段 | 必选 | 类型 | 说明 |
| --- | --- | --- | --- |
| log\_id | 是 | uint64 | 请求标识码，随机数，唯一 |
| face\_list | 是 | array | 人脸列表 |
| + face\_token | 是 | string | 人脸图片的唯一标识 |
| +ctime | 是 | string | 人脸创建时间 |

**返回示例**

{

"face\_list": [

{

"face\_token": "fid1",

"ctime": "2018-01-01 00:00:00"

},

{

"face\_token": "fid2",

"ctime": "2018-01-01 10:00:00"

}

]

}

# 10.获取用户列表

## 10.1.接口描述

用于查询指定用户组中的用户列表。

## 10.2.请求说明

**请求示例**

HTTP方法：GET

请求URL： https://aip.baidubce.com/rest/2.0/face/v3/faceset/group/getusers

URL参数：

| 参数 | 值 |
| --- | --- |
| access\_token | 通过API Key和Secret Key获取的access\_token,参考“[Access Token获取](http://ai.baidu.com/docs#/Auth)” |

Header如下：

| 参数 | 值 |
| --- | --- |
| Content-Type | application/json |

Body中放置请求参数，参数详情如下：

**请求参数**

| 参数 | 必选 | 类型 | 说明 |
| --- | --- | --- | --- |
| group\_id | 是 | string | 用户组id |
| start | 否 | uint32 | 默认值0，起始序号 |
| length | 否 | uint32 | 返回数量，默认值100，最大值1000 |

**请求代码示例**

**提示一**：使用示例代码前，请记得替换其中的示例Token、图片地址或Base64信息。

**提示二**：部分语言依赖的类或库，请在代码注释中查看下载地址。

Java

package com.baidu.ai.aip;

import com.baidu.ai.aip.utils.HttpUtil;

import com.baidu.ai.aip.utils.GsonUtils;

import java.util.\*;

/\*\*

\* 获取用户列表

\*/

public class GroupGetusers {

/\*\*

\* 重要提示代码中所需工具类

\* FileUtil,Base64Util,HttpUtil,GsonUtils请从

\* https://ai.baidu.com/file/658A35ABAB2D404FBF903F64D47C1F72

\* https://ai.baidu.com/file/C8D81F3301E24D2892968F09AE1AD6E2

\* https://ai.baidu.com/file/544D677F5D4E4F17B4122FBD60DB82B3

\* https://ai.baidu.com/file/470B3ACCA3FE43788B5A963BF0B625F3

\* 下载

\*/

public static String getUsers() {

// 请求url

String url = "https://aip.baidubce.com/rest/2.0/face/v3/faceset/group/getusers";

try {

Map<String, Object> map = new HashMap<>();

map.put("group\_id", "group1");

String param = GsonUtils.toJson(map);

// 注意这里仅为了简化编码每一次请求都去获取access\_token，线上环境access\_token有过期时间， 客户端可自行缓存，过期后重新获取。

String accessToken = "[调用鉴权接口获取的token]";

String result = HttpUtil.post(url, accessToken, "application/json", param);

System.out.println(result);

return result;

} catch (Exception e) {

e.printStackTrace();

}

return null;

}

public static void main(String[] args) {

GroupGetusers.getUsers();

}

}

## 10.3.返回说明

* 返回结果

| 字段 | 必选 | 类型 | 说明 |
| --- | --- | --- | --- |
| user\_id\_list | 是 | array | 用户ID列表 |

* 返回示例
* {
* "user\_id\_list": [
* "uid1",
* "uid2"
* ]
* }

# 11.复制用户

## 11.1.接口描述

用于将已经存在于人脸库中的用户**复制到一个新的组**。

**说明**：并不是向一个指定组内添加用户，而是直接从其它组复制用户信息 如果需要注册用户，请直接使用**人脸注册接口**

## 11.2.请求说明

**请求示例**

HTTP方法：POST

请求URL： https://aip.baidubce.com/rest/2.0/face/v3/faceset/user/copy

URL参数：

| 参数 | 值 |
| --- | --- |
| access\_token | 通过API Key和Secret Key获取的access\_token,参考“[Access Token获取](http://ai.baidu.com/docs#/Auth)” |

Header如下：

| 参数 | 值 |
| --- | --- |
| Content-Type | application/json |

Body中放置请求参数，参数详情如下：

**请求参数**

| 参数 | 必选 | 类型 | 说明 |
| --- | --- | --- | --- |
| user\_id | 是 | string | 用户id |
| src\_group\_id | 是 | string | 从指定组里复制信息 |
| dst\_group\_id | 是 | string | 需要添加用户的组id |

**请求代码示例**

**提示一**：使用示例代码前，请记得替换其中的示例Token、图片地址或Base64信息。

**提示二**：部分语言依赖的类或库，请在代码注释中查看下载地址。

undefined

## 11.3.返回说明

通过返回的error\_code判断是否成功 如失败则查看error\_msg获得具体错误信息

**返回示例**

// 正确返回值

{

"error\_code": 0,

"log\_id": 3314921889,

}

// 发生错误时返回值

{

"error\_code": 223111,

"log\_id": 3111284097,

"error\_msg": "dst group is not exist"

}

# 12.删除用户

## 12.1.接口描述

用于将用户从某个组中删除。

## 12.2.请求说明

**请求示例**

HTTP方法：POST

请求URL： https://aip.baidubce.com/rest/2.0/face/v3/faceset/user/delete

URL参数：

| 参数 | 值 |
| --- | --- |
| access\_token | 通过API Key和Secret Key获取的access\_token,参考“[Access Token获取](http://ai.baidu.com/docs#/Auth)” |

Header如下：

| 参数 | 值 |
| --- | --- |
| Content-Type | application/json |

Body中放置请求参数，参数详情如下：

**请求参数**

| 参数 | 必选 | 类型 | 说明 |
| --- | --- | --- | --- |
| group\_id | 是 | string | 用户组id(由数字、字母、下划线组成，长度限制48B) ，如传入“**@ALL**”则从所有组中删除用户 |
| user\_id | 是 | string | 用户id（由数字、字母、下划线组成），长度限制128B |

**请求代码示例**

**提示一**：使用示例代码前，请记得替换其中的示例Token、图片地址或Base64信息。

**提示二**：部分语言依赖的类或库，请在代码注释中查看下载地址。

undefined

## 12.3.返回说明

通过返回的error\_code判断是否成功 如失败则查看error\_msg获得具体错误信息：

**返回参数**

| 字段 | 必选 | 类型 | 说明 |
| --- | --- | --- | --- |
| log\_id | 是 | uint64 | 请求标识码，随机数，唯一 |

**返回示例**

// 正确返回值

{

"error\_code": 0,

"log\_id": 3314921889,

}

// 发生错误时返回值

{

"error\_code":  223103,

"log\_id": 815967402,

"error\_msg": "user is not exist"

}

# 13.创建用户组

## 13.1接口描述

用于创建一个空的用户组，如果用户组已存在 则返回错误。

## 13.2.请求说明

**请求示例**

HTTP方法：POST

请求URL： https://aip.baidubce.com/rest/2.0/face/v3/faceset/group/add

URL参数：

| 参数 | 值 |
| --- | --- |
| access\_token | 通过API Key和Secret Key获取的access\_token,参考“[Access Token获取](http://ai.baidu.com/docs#/Auth)” |

Header如下：

| 参数 | 值 |
| --- | --- |
| Content-Type | application/json |

Body中放置请求参数，参数详情如下：

**请求参数**

| 参数 | 必选 | 类型 | 说明 |
| --- | --- | --- | --- |
| group\_id | 是 | string | 用户组id，标识一组用户（由数字、字母、下划线组成），长度限制128B。 |

**请求代码示例**

**提示一**：使用示例代码前，请记得替换其中的示例Token、图片地址或Base64信息。

**提示二**：部分语言依赖的类或库，请在代码注释中查看下载地址。

Java

package com.baidu.ai.aip;

import com.baidu.ai.aip.utils.HttpUtil;

import com.baidu.ai.aip.utils.GsonUtils;

import java.util.\*;

/\*\*

\* 创建用户组

\*/

public class GroupAdd {

/\*\*

\* 重要提示代码中所需工具类

\* FileUtil,Base64Util,HttpUtil,GsonUtils请从

\* https://ai.baidu.com/file/658A35ABAB2D404FBF903F64D47C1F72

\* https://ai.baidu.com/file/C8D81F3301E24D2892968F09AE1AD6E2

\* https://ai.baidu.com/file/544D677F5D4E4F17B4122FBD60DB82B3

\* https://ai.baidu.com/file/470B3ACCA3FE43788B5A963BF0B625F3

\* 下载

\*/

public static String groupAdd() {

// 请求url

String url = "https://aip.baidubce.com/rest/2.0/face/v3/faceset/group/add";

try {

Map<String, Object> map = new HashMap<>();

map.put("group\_id", "group1");

String param = GsonUtils.toJson(map);

// 注意这里仅为了简化编码每一次请求都去获取access\_token，线上环境access\_token有过期时间， 客户端可自行缓存，过期后重新获取。

String accessToken = "[调用鉴权接口获取的token]";

String result = HttpUtil.post(url, accessToken, "application/json", param);

System.out.println(result);

return result;

} catch (Exception e) {

e.printStackTrace();

}

return null;

}

public static void main(String[] args) {

GroupAdd.groupAdd();

}

}

## 13.3.返回说明

通过返回的error\_code判断是否成功 如失败则查看error\_msg获得具体错误信息：

**返回参数**

| 字段 | 必选 | 类型 | 说明 |
| --- | --- | --- | --- |
| log\_id | 是 | uint64 | 请求标识码，随机数，唯一 |

**返回示例**

// 正确返回值

{

"error\_code": 0,

"log\_id": 3314921889,

}

// 发生错误时返回值

{

"error\_code":  223101,

"log\_id": 815967402,

"error\_msg": " group is already exist"

}

# 14.删除用户组

## 14.1.接口描述

删除用户组下所有的用户及人脸，如果组不存在 则返回错误。

## 14.2请求说明

**请求示例**

HTTP方法：POST

请求URL： https://aip.baidubce.com/rest/2.0/face/v3/faceset/group/delete

URL参数：

| 参数 | 值 |
| --- | --- |
| access\_token | 通过API Key和Secret Key获取的access\_token,参考“[Access Token获取](http://ai.baidu.com/docs#/Auth)” |

Header如下：

| 参数 | 值 |
| --- | --- |
| Content-Type | application/json |

Body中放置请求参数，参数详情如下：

**请求参数**

| 参数 | 必选 | 类型 | 说明 |
| --- | --- | --- | --- |
| group\_id | 是 | string | 用户组id，标识一组用户（由数字、字母、下划线组成），长度限48B。 |

**请求代码示例**

**提示一**：使用示例代码前，请记得替换其中的示例Token、图片地址或Base64信息。

**提示二**：部分语言依赖的类或库，请在代码注释中查看下载地址。

Java

package com.baidu.ai.aip;

import com.baidu.ai.aip.utils.HttpUtil;

import com.baidu.ai.aip.utils.GsonUtils;

import java.util.\*;

/\*\*

\* 删除用户组

\*/

public class GroupDelete {

/\*\*

\* 重要提示代码中所需工具类

\* FileUtil,Base64Util,HttpUtil,GsonUtils请从

\* https://ai.baidu.com/file/658A35ABAB2D404FBF903F64D47C1F72

\* https://ai.baidu.com/file/C8D81F3301E24D2892968F09AE1AD6E2

\* https://ai.baidu.com/file/544D677F5D4E4F17B4122FBD60DB82B3

\* https://ai.baidu.com/file/470B3ACCA3FE43788B5A963BF0B625F3

\* 下载

\*/

public static String groupDelete() {

// 请求url

String url = "https://aip.baidubce.com/rest/2.0/face/v3/faceset/group/delete";

try {

Map<String, Object> map = new HashMap<>();

map.put("group\_id", "group1");

String param = GsonUtils.toJson(map);

// 注意这里仅为了简化编码每一次请求都去获取access\_token，线上环境access\_token有过期时间， 客户端可自行缓存，过期后重新获取。

String accessToken = "[调用鉴权接口获取的token]";

String result = HttpUtil.post(url, accessToken, "application/json", param);

System.out.println(result);

return result;

} catch (Exception e) {

e.printStackTrace();

}

return null;

}

public static void main(String[] args) {

GroupDelete.groupDelete();

}

}

## 14.3.返回说明

通过返回的error\_code判断是否成功 如失败则查看error\_msg获得具体错误信息：

**返回参数**

| 字段 | 必选 | 类型 | 说明 |
| --- | --- | --- | --- |
| log\_id | 是 | uint64 | 请求标识码，随机数，唯一 |

**返回示例**

// 正确返回值

{

"error\_code":0,

"log\_id": 3314921889,

}

// 发生错误时返回值

{

"error\_code": 223100,

"log\_id": 815967402,

"error\_msg": " group is not exist"

}

# 15.组列表查询

## 15.1.接口描述

用于查询用户组的列表。

## 15.2.请求说明

**请求示例**

HTTP方法：POST

请求URL： https://aip.baidubce.com/rest/2.0/face/v3/faceset/group/getlist

URL参数：

| 参数 | 值 |
| --- | --- |
| access\_token | 通过API Key和Secret Key获取的access\_token,参考“[Access Token获取](http://ai.baidu.com/docs#/Auth)” |

Header如下：

| 参数 | 值 |
| --- | --- |
| Content-Type | application/json |

Body中放置请求参数，参数详情如下：

**请求参数**

| 参数 | 必选 | 类型 | 说明 |
| --- | --- | --- | --- |
| start | 否 | uint32 | 默认值0，起始序号 |
| length | 否 | uint32 | 返回数量，默认值100，最大值1000 |

**请求代码示例**

**提示一**：使用示例代码前，请记得替换其中的示例Token、图片地址或Base64信息。

**提示二**：部分语言依赖的类或库，请在代码注释中查看下载地址。

Java

package com.baidu.ai.aip;

import com.baidu.ai.aip.utils.HttpUtil;

import com.baidu.ai.aip.utils.GsonUtils;

import java.util.\*;

/\*\*

\* 组列表查询

\*/

public class GroupGetlist {

/\*\*

\* 重要提示代码中所需工具类

\* FileUtil,Base64Util,HttpUtil,GsonUtils请从

\* https://ai.baidu.com/file/658A35ABAB2D404FBF903F64D47C1F72

\* https://ai.baidu.com/file/C8D81F3301E24D2892968F09AE1AD6E2

\* https://ai.baidu.com/file/544D677F5D4E4F17B4122FBD60DB82B3

\* https://ai.baidu.com/file/470B3ACCA3FE43788B5A963BF0B625F3

\* 下载

\*/

public static String GroupGetlist() {

// 请求url

String url = "https://aip.baidubce.com/rest/2.0/face/v3/faceset/group/getlist";

try {

Map<String, Object> map = new HashMap<>();

map.put("start", 0);

map.put("length", 100);

String param = GsonUtils.toJson(map);

// 注意这里仅为了简化编码每一次请求都去获取access\_token，线上环境access\_token有过期时间， 客户端可自行缓存，过期后重新获取。

String accessToken = "[调用鉴权接口获取的token]";

String result = HttpUtil.post(url, accessToken, "application/json", param);

System.out.println(result);

return result;

} catch (Exception e) {

e.printStackTrace();

}

return null;

}

public static void main(String[] args) {

GroupGetlist.GroupGetlist();

}

}

## 15.3.返回说明

**返回参数**

* 返回结果

| 字段 | 必选 | 类型 | 说明 |
| --- | --- | --- | --- |
| group\_id\_list | 是 | array | group |

* 返回示例
* {
* "group\_id\_list": [
* "gid1",
* "gid2"
* ]
* }

## 15.4.错误码

请参考[人脸识别错误码](http://ai.baidu.com/docs#/Face-ErrorCode-V3/top)

# 16.身份验证

**人脸识别接口分为V2和V3两个版本，本文档为V3版本接口的说明文档，请确认您在百度云后台获得的是V3版本接口权限，再来阅读本文档。**

辨别接口版本的方法是：在百度云后台进入【应用列表】，点击【应用名称】，在【API列表】中可以看到【请求地址】，若请求地址中带有【v3】标识，则您具有的是v3权限，可以阅读本文档；若请求地址中带有【v2】标识，则您具有的是v2权限，应该去阅读v2文档。 如果您对文档内容有任何疑问，可以通过以下几种方式联系我们：

* 在百度云控制台内 [提交工单](http://ticket.bce.baidu.com/#/ticket/create)，咨询问题类型请选择**人工智能服务**；
* 如有需要讨论的疑问，欢迎进入 [AI社区](http://ai.baidu.com/forum/topic/list/165) 与其他开发者们一同交流。

## 16.1.能力介绍

**业务能力**

* **质量检测（可选）**：判断图片中是否包含人脸，以及人脸在姿态、遮挡、模糊、光照等方面是否符合识别条件；
* **活体检测（可选）**：基于图片中的破绽分析，判断其中的人脸是否为**二次翻拍**（举例：如用户A用手机拍摄了一张包含人脸的图片一，用户B翻拍了图片一得到了图片二，并用图片二伪造成用户A去进行识别操作，这种情况普遍发生在金融开户、实名认证等环节。）；
* **公安验证（必选）**：基于姓名和身份证号，调取公民身份证小图（源自公安系统），将当前获取的人脸图片，与此证件小图进行对比，得出比对分数，并基于此进行业务判断是否为同一人。由于公安系统小图，具有最权威的身份证明作用，故对用户本人的验证结果可信度也最为合理。

**业务逻辑**

* 上述三项能力为顺序串行验证，接口默认返回公安身份对比分值，质量检测和活体检测为可选项。如选择了这两项，则验证顺序为人脸质量检测->活体检测->公安身份验证。您也可以根据业务场景，选择这两项中的某一项或都不选择。
* **如选择了前两个环节，则有任意一个条件不通过，则整个请求流程终止，并返回错误码，描述具体的不符合信息**。
* 基于此顺序串行验证逻辑，可以避免大量不符合条件的请求流转到公安验证，节约您的成本。

**推荐阈值**

* 此接口使用的对比算法，针对带水纹证件照采用了专项的模型处理，可保证水纹信息的影响降到尽可能低。
* 如比对成功，最终返回的有效数据为一个**对比分值**，在0~1之间，您可以设定具体的阈值来判断是否验证通过。
* **人证相似度的推荐阈值为0.8，对应的误识率为万分之一**。

**计费逻辑**

* 前两个环节图片不符合校验规则，会以error\_code形式反馈，属于正向业务判断，这两个环节的请求全部免费。
* 真正请求到公安验证这步并返回result，才会进行计费。或者您也可以理解为：**只要是错误码情况，全部都不计费**.
* 按次计费，单价：**0.6元/次**。

**温馨提示**：用户可能会因网络原因等原因重复请求，对于**每隔5分钟**内的重复请求（image、name、id\_card\_number相同），**只计费一次**。

## 16.2.调用方式

**请求URL数据格式**

向API服务地址使用POST发送请求，必须在URL中带上参数access\_token，可通过后台的API Key和Secret Key生成，具体方式请参考“[Access Token获取](http://ai.baidu.com/docs#/Auth)”。

**获取access\_token的示例代码**

Java

package com.baidu.ai.aip.auth;

import org.json.JSONObject;

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.net.HttpURLConnection;

import java.net.URL;

import java.util.List;

import java.util.Map;

/\*\*

\* 获取token类

\*/

public class AuthService {

/\*\*

\* 获取权限token

\* @return 返回示例：

\* {

\* "access\_token": "24.460da4889caad24cccdb1fea17221975.2592000.1491995545.282335-1234567",

\* "expires\_in": 2592000

\* }

\*/

public static String getAuth() {

// 官网获取的 API Key 更新为你注册的

String clientId = "百度云应用的AK";

// 官网获取的 Secret Key 更新为你注册的

String clientSecret = "百度云应用的SK";

return getAuth(clientId, clientSecret);

}

/\*\*

\* 获取API访问token

\* 该token有一定的有效期，需要自行管理，当失效时需重新获取.

\* @param ak - 百度云官网获取的 API Key

\* @param sk - 百度云官网获取的 Securet Key

\* @return assess\_token 示例：

\* "24.460da4889caad24cccdb1fea17221975.2592000.1491995545.282335-1234567"

\*/

public static String getAuth(String ak, String sk) {

// 获取token地址

String authHost = "https://aip.baidubce.com/oauth/2.0/token?";

String getAccessTokenUrl = authHost

// 1. grant\_type为固定参数

+ "grant\_type=client\_credentials"

// 2. 官网获取的 API Key

+ "&client\_id=" + ak

// 3. 官网获取的 Secret Key

+ "&client\_secret=" + sk;

try {

URL realUrl = new URL(getAccessTokenUrl);

// 打开和URL之间的连接

HttpURLConnection connection = (HttpURLConnection) realUrl.openConnection();

connection.setRequestMethod("GET");

connection.connect();

// 获取所有响应头字段

Map<String, List<String>> map = connection.getHeaderFields();

// 遍历所有的响应头字段

for (String key : map.keySet()) {

System.err.println(key + "--->" + map.get(key));

}

// 定义 BufferedReader输入流来读取URL的响应

BufferedReader in = new BufferedReader(new InputStreamReader(connection.getInputStream()));

String result = "";

String line;

while ((line = in.readLine()) != null) {

result += line;

}

/\*\*

\* 返回结果示例

\*/

System.err.println("result:" + result);

JSONObject jsonObject = new JSONObject(result);

String access\_token = jsonObject.getString("access\_token");

return access\_token;

} catch (Exception e) {

System.err.printf("获取token失败！");

e.printStackTrace(System.err);

}

return null;

}

}

**注意**：access\_token的有效期为30天，**切记需要每30天进行定期更换，或者每次请求都拉取新token**；

例如此接口，使用HTTPS POST发送：

https://aip.baidubce.com/rest/2.0/face/v3/person/verify?access\_token=24.f9ba9c5341b67688ab4added8bc91dec.2592000.1485570332.282335-8574074

POST中Body的参数，按照下方请求参数说明选择即可。

**提示**：如果您为百度云老用户，正在使用其他非AI的服务，可以参考[百度云AKSK鉴权方式](https://cloud.baidu.com/doc/Reference/AuthenticationMechanism.html)发送请求，虽然**请求方式**和**鉴权方法**和本文所介绍的不同，但请求参数和返回结果一致。

## 16.3.请求说明

**注意事项**：

* **请求体格式化**：Content-Type为application/json，通过json格式化请求体。
* **Base64编码**：请求的图片需经过Base64编码，图片的base64编码指将图片数据编码成一串字符串，使用该字符串代替图像地址。您可以首先得到图片的二进制，然后用Base64格式编码即可。需要注意的是，图片的base64编码是不包含图片头的，如data:image/jpg;base64,
* **图片格式**：现支持PNG、JPG、JPEG、BMP，**不支持GIF图片**
* **人脸识别接口分为V2和V3两个版本，本文档为V3版本接口的说明文档，请确认您在百度云后台获得的是V3版本接口权限，再来阅读本文档。**

辨别接口版本的方法是：在百度云后台进入【应用列表】，点击【应用名称】，在【API列表】中可以看到【请求地址】，若请求地址中带有【v3】标识，则您具有的是v3权限，可以阅读本文档；若请求地址中带有【v2】标识，则您具有的是v2权限，应该去阅读v2文档。

**请求示例**

HTTP方法：POST

请求URL： https://aip.baidubce.com/rest/2.0/face/v3/person/verify

URL参数：

| 参数 | 值 |
| --- | --- |
| access\_token | 通过API Key和Secret Key获取的access\_token,参考“[Access Token获取](http://ai.baidu.com/docs#Beginner-Auth)” |

Header：

| 参数 | 值 |
| --- | --- |
| Content-Type | application/json |

Body中放置请求参数，参数详情如下：

**请求参数**

| 参数 | 必选 | 类型 | 说明 |
| --- | --- | --- | --- |
| image | 是 | string | 图片信息(**总数据大小应小于10M**)，图片上传方式根据image\_type来判断 |
| image\_type | 是 | string | 图片类型 **BASE64**:图片的base64值，base64编码后的图片数据，需urlencode，编码后的图片大小不超过2M； **URL**:图片的 URL地址( 可能由于网络等原因导致下载图片时间过长)\*\*； **FACE\_TOKEN**: 人脸图片的唯一标识，调用人脸检测接口时，会为每个人脸图片赋予一个唯一的FACE\_TOKEN，同一张图片多次检测得到的FACE\_TOKEN是同一个。 |
| id\_card\_number | 是 | string | 身份证号码 |
| name | 是 | string | 姓名（注：需要进行utf8编码） |
| quality\_control | 否 | string | 图片质量控制 **NONE**: 不进行控制 **LOW**:较低的质量要求 **NORMAL**: 一般的质量要求 **HIGH**: 较高的质量要求 **默认 NONE** |
| liveness\_control | 否 | uint32 | 活体检测控制 **NONE**: 不进行控制 **LOW**:较低的活体要求(高通过率 低攻击拒绝率) **NORMAL**: 一般的活体要求(平衡的攻击拒绝率, 通过率) **HIGH**: 较高的活体要求(高攻击拒绝率 低通过率) **默认NONE** |

**请求代码示例**

**提示一**：使用示例代码前，请记得替换其中的示例Token、图片地址或Base64信息。

**提示二**：部分语言依赖的类或库，请在代码注释中查看下载地址。

Java

package com.baidu.ai.aip;

import com.baidu.ai.aip.utils.HttpUtil;

import com.baidu.ai.aip.utils.GsonUtils;

import java.util.\*;

/\*\*

\* 身份验证

\*/

public class PersonVerify {

/\*\*

\* 重要提示代码中所需工具类

\* FileUtil,Base64Util,HttpUtil,GsonUtils请从

\* https://ai.baidu.com/file/658A35ABAB2D404FBF903F64D47C1F72

\* https://ai.baidu.com/file/C8D81F3301E24D2892968F09AE1AD6E2

\* https://ai.baidu.com/file/544D677F5D4E4F17B4122FBD60DB82B3

\* https://ai.baidu.com/file/470B3ACCA3FE43788B5A963BF0B625F3

\* 下载

\*/

public static String personverify() {

// 请求url

String url = "https://aip.baidubce.com/rest/2.0/face/v3/person/verify";

try {

Map<String, Object> map = new HashMap<>();

map.put("image", "sfasq35sadvsvqwr5q...");

map.put("image\_type", "BASE64");

map.put("id\_card\_number", "1234");

map.put("liveness\_control", "HIGH");

map.put("name", "张三");

map.put("quality\_control", "LOW");

String param = GsonUtils.toJson(map);

// 注意这里仅为了简化编码每一次请求都去获取access\_token，线上环境access\_token有过期时间， 客户端可自行缓存，过期后重新获取。

String accessToken = "【调用鉴权接口获取的token】";

String result = HttpUtil.post(url, accessToken, "application/json", param);

System.out.println(result);

return result;

} catch (Exception e) {

e.printStackTrace();

}

return null;

}

public static void main(String[] args) {

PersonVerify.personverify();

}

}

## 16.4.返回说明

**返回参数**

| 参数 | 必须 | 类型 | 说明 |
| --- | --- | --- | --- |
| log\_id | 是 | uint64 | 日志id |
| score | 是 | float | 与公安小图相似度可能性，用于验证生活照与公安小图是否为同一人，有正常分数时为[0~100]，推荐阈值80，超过即判断为同一人 |

**返回示例**

{

"score": 44.3,

}

* **质量控制参数说明**

不同的控制度下所对应的质量控制阈值，如果检测出来的质量信息某一项不符合控制阈值的要求，则会返回错误信息。

| 控制度 | left\_eye | right\_eye | nose | mouth | left\_cheek | right\_cheek | chin\_contour | illumination | blurdegree | completeness |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| LOW | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 0.8 | 20 | 0.8 | 0 |
| NORMAL | 0.6 | 0.6 | 0.6 | 0.6 | 0.6 | 0.6 | 0.6 | 40 | 0.6 | 0 |
| HIGH | 0.2 | 0.2 | 0.2 | 0.2 | 0.2 | 0.2 | 0.2 | 100 | 0.2 | 1 |

**活体控制参数说明**

不同的控制度下所对应的活体控制阈值，如果检测出来的活体分数小于控制阈值，则会返回错误信息。

| 控制度 | 阈值 | 说明 |
| --- | --- | --- |
| LOW | 0.05 | 万分之一活体误拒率 |
| NORMAL | 0.3 | 千分之一活体误拒率 |
| HIGH | 0.9 | 百分之一活体误拒率 |

误拒率: 把**真人识别为假人**的概率. 阈值越高，安全性越高, 要求也就越高, 对应的误识率就越高

## 16.5.错误码

请参考[人脸识别错误码](http://ai.baidu.com/docs#/Face-ErrorCode-V3/top)

# 17.在线活体检测

**人脸识别接口分为V2和V3两个版本，本文档为V3版本接口的说明文档，请确认您在百度云后台获得的是V3版本接口权限，再来阅读本文档。**

辨别接口版本的方法是：在百度云后台进入【应用列表】，点击【应用名称】，在【API列表】中可以看到【请求地址】，若请求地址中带有【v3】标识，则您具有的是v3权限，可以阅读本文档；若请求地址中带有【v2】标识，则您具有的是v2权限，应该去阅读v2文档。 如果您对文档内容有任何疑问，可以通过以下几种方式联系我们：

* 在百度云控制台内 [提交工单](http://ticket.bce.baidu.com/#/ticket/create)，咨询问题类型请选择**人工智能服务**；
* 如有需要讨论的疑问，欢迎进入 [AI社区](http://ai.baidu.com/forum/topic/list/165) 与其他开发者们一同交流。

## 17.1能力介绍

**接口能力**

* **人脸基础信息**：包括人脸框位置，人脸空间旋转角度，人脸置信度等信息。
* **人脸质量检测**：判断人脸的遮挡、光照、模糊度、完整度等质量信息。可用于判断上传的人脸是否符合标准。
* **基于图片的活体检测**：基于单张图片，判断图片中的人脸是否为二次翻拍（举例：如用户A用手机拍摄了一张包含人脸的图片一，用户B翻拍了图片一得到了图片二，并用图片二伪造成用户A去进行识别操作，这种情况普遍发生在金融开户、实名认证等环节）。此能力可用于H5场景下的一些人脸采集场景中，增加人脸注册的安全性和真实性。

## 17.2.调用方式

**请求URL数据格式**

向API服务地址使用POST发送请求，必须在URL中带上参数access\_token，可通过后台的API Key和Secret Key生成，具体方式请参考“[Access Token获取](http://ai.baidu.com/docs#/Auth)”。

**获取access\_token的示例代码**

Java

package com.baidu.ai.aip.auth;

import org.json.JSONObject;

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.net.HttpURLConnection;

import java.net.URL;

import java.util.List;

import java.util.Map;

/\*\*

\* 获取token类

\*/

public class AuthService {

/\*\*

\* 获取权限token

\* @return 返回示例：

\* {

\* "access\_token": "24.460da4889caad24cccdb1fea17221975.2592000.1491995545.282335-1234567",

\* "expires\_in": 2592000

\* }

\*/

public static String getAuth() {

// 官网获取的 API Key 更新为你注册的

String clientId = "百度云应用的AK";

// 官网获取的 Secret Key 更新为你注册的

String clientSecret = "百度云应用的SK";

return getAuth(clientId, clientSecret);

}

/\*\*

\* 获取API访问token

\* 该token有一定的有效期，需要自行管理，当失效时需重新获取.

\* @param ak - 百度云官网获取的 API Key

\* @param sk - 百度云官网获取的 Securet Key

\* @return assess\_token 示例：

\* "24.460da4889caad24cccdb1fea17221975.2592000.1491995545.282335-1234567"

\*/

public static String getAuth(String ak, String sk) {

// 获取token地址

String authHost = "https://aip.baidubce.com/oauth/2.0/token?";

String getAccessTokenUrl = authHost

// 1. grant\_type为固定参数

+ "grant\_type=client\_credentials"

// 2. 官网获取的 API Key

+ "&client\_id=" + ak

// 3. 官网获取的 Secret Key

+ "&client\_secret=" + sk;

try {

URL realUrl = new URL(getAccessTokenUrl);

// 打开和URL之间的连接

HttpURLConnection connection = (HttpURLConnection) realUrl.openConnection();

connection.setRequestMethod("GET");

connection.connect();

// 获取所有响应头字段

Map<String, List<String>> map = connection.getHeaderFields();

// 遍历所有的响应头字段

for (String key : map.keySet()) {

System.err.println(key + "--->" + map.get(key));

}

// 定义 BufferedReader输入流来读取URL的响应

BufferedReader in = new BufferedReader(new InputStreamReader(connection.getInputStream()));

String result = "";

String line;

while ((line = in.readLine()) != null) {

result += line;

}

/\*\*

\* 返回结果示例

\*/

System.err.println("result:" + result);

JSONObject jsonObject = new JSONObject(result);

String access\_token = jsonObject.getString("access\_token");

return access\_token;

} catch (Exception e) {

System.err.printf("获取token失败！");

e.printStackTrace(System.err);

}

return null;

}

}

**注意**：access\_token的有效期为30天，**切记需要每30天进行定期更换，或者每次请求都拉取新token**；

例如此接口，使用HTTPS POST发送：

https://aip.baidubce.com/rest/2.0/face/v3/faceverify?access\_token=24.f9ba9c5341b67688ab5added8bc91dec.2592000.1485570332.282335-8574075

POST中Body的参数，按照下方请求参数说明选择即可。

**提示**：如果您为百度云老用户，正在使用其他非AI的服务，可以参考[百度云AKSK鉴权方式](https://cloud.baidu.com/doc/Reference/AuthenticationMechanism.html)发送请求，虽然**请求方式**和**鉴权方法**和本文所介绍的不同，但请求参数和返回结果一致。

## 17.3.请求说明

**注意事项**：

* **请求体格式化**：Content-Type为application/json，通过json格式化请求体。
* **Base64编码**：请求的图片需经过Base64编码，图片的base64编码指将图片数据编码成一串字符串，使用该字符串代替图像地址。您可以首先得到图片的二进制，然后用Base64格式编码即可。需要注意的是，图片的base64编码是不包含图片头的，如data:image/jpg;base64,
* **图片格式**：现支持PNG、JPG、JPEG、BMP，**不支持GIF图片**
* **人脸识别接口分为V2和V3两个版本，本文档为V3版本接口的说明文档，请确认您在百度云后台获得的是V3版本接口权限，再来阅读本文档。**

辨别接口版本的方法是：在百度云后台进入【应用列表】，点击【应用名称】，在【API列表】中可以看到【请求地址】，若请求地址中带有【v3】标识，则您具有的是v3权限，可以阅读本文档；若请求地址中带有【v2】标识，则您具有的是v2权限，应该去阅读v2文档。

**请求示例**

HTTP方法：POST

请求URL： https://aip.baidubce.com/rest/2.0/face/v3/faceverify

URL参数：

| 参数 | 值 |
| --- | --- |
| access\_token | 通过API Key和Secret Key获取的access\_token,参考“[Access Token获取](http://ai.baidu.com/docs#Beginner-Auth)” |

Header：

| 参数 | 值 |
| --- | --- |
| Content-Type | application/json |

Body中放置请求参数，参数详情如下：

**请求参数**

| 参数 | 是否必选 | 类型 | 说明 |
| --- | --- | --- | --- |
| image | 是 | string | 图片信息(总数据大小应小于10M)，图片上传方式根据image\_type来判断； **可以上传同一个用户的1张、3张或8张图片来进行活体判断，注：后端会选择每组照片中的最高分数作为整体分数。** |
| image\_type | 是 | string | 图片类型 **BASE64**:图片的base64值，base64编码后的图片数据，需urlencode，编码后的图片大小不超过2M； **URL**:图片的 URL地址( 可能由于网络等原因导致下载图片时间过长)； **FACE\_TOKEN**: 人脸图片的唯一标识，调用人脸检测接口时，会为每个人脸图片赋予一个唯一的FACE\_TOKEN，同一张图片多次检测得到的FACE\_TOKEN是同一个。 |
| face\_field | 否 | string | 包括**age,beauty,expression,faceshape,gender,glasses,landmark,race,quality,facetype信息**，逗号分隔，默认只返回face\_token、活体数、人脸框、概率和旋转角度 |

**请求示例代码**

**提示一**：使用示例代码前，请记得替换其中的示例Token、图片地址或Base64信息。

**提示二**：部分语言依赖的类或库，请在代码注释中查看下载地址。

Java

package com.baidu.ai.aip;

import com.baidu.ai.aip.utils.HttpUtil;

import com.baidu.ai.aip.utils.GsonUtils;

import java.util.\*;

/\*\*

\* 在线活体检测

\*/

public class FaceVerify {

/\*\*

\* 重要提示代码中所需工具类

\* FileUtil,Base64Util,HttpUtil,GsonUtils请从

\* https://ai.baidu.com/file/658A35ABAB2D404FBF903F64D47C1F72

\* https://ai.baidu.com/file/C8D81F3301E24D2892968F09AE1AD6E2

\* https://ai.baidu.com/file/544D677F5D4E4F17B4122FBD60DB82B3

\* https://ai.baidu.com/file/470B3ACCA3FE43788B5A963BF0B625F3

\* 下载

\*/

public static String faceVerify() {

// 请求url

String url = "https://aip.baidubce.com/rest/2.0/face/v3/faceverify";

try {

String param = GsonUtils.toJson(map);

// 注意这里仅为了简化编码每一次请求都去获取access\_token，线上环境access\_token有过期时间， 客户端可自行缓存，过期后重新获取。

String accessToken = "[调用鉴权接口获取的token]";

String result = HttpUtil.post(url, accessToken, "application/json", param);

System.out.println(result);

return result;

} catch (Exception e) {

e.printStackTrace();

}

return null;

}

public static void main(String[] args) {

FaceVerify.faceVerify();

}

}

### 17.3.1.返回参数

| 参数 | 类型 | 是否必须 | 说明 |
| --- | --- | --- | --- |
| face\_liveness | 是 | float | 活体分数值 |
| thresholds | 是 | array | 由服务端返回最新的阈值数据（随着模型的优化，阈值可能会变化），将此参数与返回的face\_liveness进行比较，可以作为活体判断的依据。 frr\_1e-4：万分之一误识率的阈值；frr\_1e-3：千分之一误识率的阈值；frr\_1e-2：百分之一误识率的阈值。误识率越低，准确率越高，相应的拒绝率也越高 |
| face\_list | 是 | array | 每张图片的详细信息描述，如果只上传一张图片，则只返回一个结果。 |
| +face\_token | 是 | string | 人脸图片的唯一标识 |
| +location | 是 | array | 人脸在图片中的位置 |
| ++left | 是 | double | 人脸区域离左边界的距离 |
| ++top | 是 | double | 人脸区域离上边界的距离 |
| ++width | 是 | double | 人脸区域的宽度 |
| ++height | 是 | double | 人脸区域的高度 |
| ++rotation | 是 | int64 | 人脸框相对于竖直方向的顺时针旋转角，[-180,180] |
| +face\_probability | 是 | double | 人脸置信度，范围【0~1】，代表这是一张人脸的概率，0最小、1最大。 |
| +angel | 是 | array | 人脸旋转角度参数 |
| ++yaw | 是 | double | 三维旋转之左右旋转角[-90(左), 90(右)] |
| ++pitch | 是 | double | 三维旋转之俯仰角度[-90(上), 90(下)] |
| ++roll | 是 | double | 平面内旋转角[-180(逆时针), 180(顺时针)] |
| +age | 否 | double | 年龄 ，当**face\_field包含age时返回** |
| +beauty | 否 | int64 | 美丑打分，范围0-100，越大表示越美。当**face\_fields包含beauty时返回** |
| +expression | 否 | array | 表情，当 **face\_field包含expression时返回** |
| ++type | 否 | string | **none**:不笑；**smile**:微笑；**laugh**:大笑 |
| ++probability | 否 | double | 表情置信度，范围【0~1】，0最小、1最大。 |
| +face\_shape | 否 | array | 脸型，当**face\_field包含faceshape时返回** |
| ++type | 否 | double | **square**: 正方形 **triangle**:三角形 **oval**: 椭圆 **heart**: 心形 **round**: 圆形 |
| ++probability | 否 | double | 置信度，范围【0~1】，代表这是人脸形状判断正确的概率，0最小、1最大。 |
| +gender | 否 | array | 性别，**face\_field包含gender时返回** |
| ++type | 否 | string | male:**男性** female:**女性** |
| ++probability | 否 | double | 性别置信度，范围【0~1】，0代表概率最小、1代表最大。 |
| +glasses | 否 | array | 是否带眼镜，**face\_field包含glasses时返回** |
| ++type | 否 | string | **none**:无眼镜，**common**:普通眼镜，**sun**:墨镜 |
| ++probability | 否 | double | 眼镜置信度，范围【0~1】，0代表概率最小、1代表最大。 |
| +race | 否 | array | 人种 **face\_field包含race时返回** |
| ++type | 否 | string | **yellow**: 黄种人 **white**: 白种人 **black**:黑种人 **arabs**: **阿拉伯人** |
| ++probability | 否 | double | 人种置信度，范围【0~1】，0代表概率最小、1代表最大。 |
| +face\_type | 否 | array | 真实人脸/卡通人脸 **face\_field包含facetype时返回** |
| ++type | 否 | string | **human**: 真实人脸 **cartoon**: 卡通人脸 |
| ++probability | 否 | double | 人脸类型判断正确的置信度，范围【0~1】，0代表概率最小、1代表最大。 |
| +landmark | 否 | array | 4个关键点位置，左眼中心、右眼中心、鼻尖、嘴中心。**face\_field包含landmark时返回** |
| +landmark72 | 否 | array | 72个特征点位置 **face\_field包含landmark时返回** |
| +quality | 否 | array | 人脸质量信息。**face\_field包含quality时返回** |
| ++occlusion | 否 | array | 人脸各部分遮挡的概率，范围[0~1]，0表示完整，1表示不完整 |
| +++left\_eye | 否 | double | 左眼遮挡比例，[0-1] ， 1表示完全遮挡 |
| +++right\_eye | 否 | double | 右眼遮挡比例，[0-1] ， 1表示完全遮挡 |
| +++nose | 否 | double | 鼻子遮挡比例，[0-1] ， 1表示完全遮挡 |
| +++mouth | 否 | double | 嘴巴遮挡比例，[0-1] ， 1表示完全遮挡 |
| +++left\_cheek | 否 | double | 左脸颊遮挡比例，[0-1] ， 1表示完全遮挡 |
| +++right\_cheek | 否 | double | 右脸颊遮挡比例，[0-1] ， 1表示完全遮挡 |
| +++chin | 否 | double | 下巴遮挡比例，，[0-1] ， 1表示完全遮挡 |
| ++blur | 否 | double | 人脸模糊程度，范围[0~1]，0表示清晰，1表示模糊 |
| ++illumination | 否 | double | 取值范围在[0~255], 表示脸部区域的光照程度 越大表示光照越好 |
| ++completeness | 否 | int64 | 人脸完整度，0或1, 0为人脸溢出图像边界，1为人脸都在图像边界内 |

### 17.3.2.返回示例

{

"thresholds": {

"frr\_1e-4": 0.05, //万分之一误拒率的阈值

"frr\_1e-3": 0.3, //千分之一误拒率的阈值

"frr\_1e-2": 0.9 //百分之一误拒率的阈值

},

"face\_liveness": 0.05532243927,

"face\_list": [

{

"face\_token": "df46f7c7db4aa09a093c26fb8d1a8d44",

"location": {

"left": 328.9026489,

"top": 97.16340637,

"width": 162,

"height": 154,

"rotation": 32

},

"face\_probability": 1,

"angle": {

"yaw": 10.16196251,

"pitch": 2.244354248,

"roll": 33.82199097

},

"liveness": {

"faceliveness": 0.004187555984,

"livemapscore": 0.04492170034

},

"age": 23,

"beauty": 20.23693275

},

{

"face\_token": "901d2c64274fccd687d311a6e6110a01",

"location": {

"left": 411.4876404,

"top": 166.3593445,

"width": 329,

"height": 308,

"rotation": 45

},

"face\_probability": 0.9194830656,

"angle": {

"yaw": -1.716423035,

"pitch": 7.344647408,

"roll": 45.79914856

},

"liveness": {

"faceliveness": 0.0001665892196,

"livemapscore": 0.001787073661

},

"age": 23,

"beauty": 12.6438179

},

{

"face\_token": "7d57e36981c48b4946eb97c8d838b02a",

"location": {

"left": 161.4559937,

"top": 199.8726501,

"width": 218,

"height": 201,

"rotation": -1

},

"face\_probability": 1,

"angle": {

"yaw": -8.187754631,

"pitch": 6.973727226,

"roll": -1.25429821

},

"liveness": {

"faceliveness": 0.02942637168,

"livemapscore": 0.05532243927

},

"age": 23,

"beauty": 8.20657444

}

]

}

## 17.4.错误码

请参考[人脸识别错误码](http://ai.baidu.com/docs#/Face-ErrorCode-V3/top)

# 18.H5视频活体检测

如果您对文档内容有任何疑问，可以通过以下几种方式联系我们：

* 在百度云控制台内 [提交工单](http://ticket.bce.baidu.com/#/ticket/create)，咨询问题类型请选择**人工智能服务**；
* 如有需要讨论的疑问，欢迎进入 [AI社区](http://ai.baidu.com/forum/topic/list/165) 与其他开发者们一同交流。

## 18.1.能力介绍

**业务能力**

H5视频活体检测产品，是由两个接口组合而成，主要用于在H5场景下，通过用户新录制并上传一个视频，来进行活体检测的判断。相对于APP有动作校验+静默图片活体、静默图片活体这两种方式，H5视频活体方案比APP方案更加灵活，同时比单张图片活体方式更加安全。其主要功能如下所示：

* 语音校验码：为防止用户提交非当前操作的视频，在录制视频时，随机分配一个数字，用户需要读出这个数字，声音存于视频当中，并在后续识别时校验，以判断是否为此次会话。
* 视频多帧活体检测：录制并上传的视频，会在云端进行随机抽帧分析，并得出最终的活体检测分数。

以上两项能力，分为两个接口，使用顺序为语音校验码->视频多帧校验，具体调用逻辑可以参考我们的。Demo体验地址（电脑端用谷歌、火狐浏览器）：  
Mobile：https://ai.baidu.com/face-verify/mobile.html  
PC：https://ai.baidu.com/face-verify/pc.html  
[Demo源代码下载地址](https://pan.baidu.com/s/1cOWwyBSkir6nR9v0isteUw),下载包中包含前端、服务端的源代码和部署说明。

**主要适用场景**

* 微信服务号：用于对操作用户真实性要求严格的场景，用于依托于微信服务号的金融开户、实名认证、账户信息变更二次验证等服务。
* APP内Webview：对于如Cordova架构开发的APP，或者APP内变更频繁的身份信息页等情况，可以采用此方案完成活体检测。
* 浏览器：如果仅是一个H5宣传页，或者Wap版本网页等，可以通过此方法快速集成更加安全的活体检测功能。

**此方案的优劣势**

* 优势：相对于APP有动作校验、单张图片静默判断，此方法在没有APP情况下，可以更快速、轻量级地实现活体检测，同时保障一定安全性。
* 劣势：由于视频较大，上传时间可能较长，另由于不同手机的浏览器内核差异较大，容易出现兼容性问题。

## 18.2.调用方式

**请求URL数据格式**

向API服务地址使用POST发送请求，必须在URL中带上参数access\_token，可通过后台的API Key和Secret Key生成，具体方式请参考“[Access Token获取](http://ai.baidu.com/docs#/Auth)”。

**获取access\_token的示例代码**

Java

package com.baidu.ai.aip.auth;

import org.json.JSONObject;

import java.io.BufferedReader;

import java.io.InputStreamReader;

import java.net.HttpURLConnection;

import java.net.URL;

import java.util.List;

import java.util.Map;

/\*\*

\* 获取token类

\*/

public class AuthService {

/\*\*

\* 获取权限token

\* @return 返回示例：

\* {

\* "access\_token": "24.460da4889caad24cccdb1fea17221975.2592000.1491995545.282335-1234567",

\* "expires\_in": 2592000

\* }

\*/

public static String getAuth() {

// 官网获取的 API Key 更新为你注册的

String clientId = "百度云应用的AK";

// 官网获取的 Secret Key 更新为你注册的

String clientSecret = "百度云应用的SK";

return getAuth(clientId, clientSecret);

}

/\*\*

\* 获取API访问token

\* 该token有一定的有效期，需要自行管理，当失效时需重新获取.

\* @param ak - 百度云官网获取的 API Key

\* @param sk - 百度云官网获取的 Securet Key

\* @return assess\_token 示例：

\* "24.460da4889caad24cccdb1fea17221975.2592000.1491995545.282335-1234567"

\*/

public static String getAuth(String ak, String sk) {

// 获取token地址

String authHost = "https://aip.baidubce.com/oauth/2.0/token?";

String getAccessTokenUrl = authHost

// 1. grant\_type为固定参数

+ "grant\_type=client\_credentials"

// 2. 官网获取的 API Key

+ "&client\_id=" + ak

// 3. 官网获取的 Secret Key

+ "&client\_secret=" + sk;

try {

URL realUrl = new URL(getAccessTokenUrl);

// 打开和URL之间的连接

HttpURLConnection connection = (HttpURLConnection) realUrl.openConnection();

connection.setRequestMethod("GET");

connection.connect();

// 获取所有响应头字段

Map<String, List<String>> map = connection.getHeaderFields();

// 遍历所有的响应头字段

for (String key : map.keySet()) {

System.err.println(key + "--->" + map.get(key));

}

// 定义 BufferedReader输入流来读取URL的响应

BufferedReader in = new BufferedReader(new InputStreamReader(connection.getInputStream()));

String result = "";

String line;

while ((line = in.readLine()) != null) {

result += line;

}

/\*\*

\* 返回结果示例

\*/

System.err.println("result:" + result);

JSONObject jsonObject = new JSONObject(result);

String access\_token = jsonObject.getString("access\_token");

return access\_token;

} catch (Exception e) {

System.err.printf("获取token失败！");

e.printStackTrace(System.err);

}

return null;

}

}

**注意**：access\_token的有效期为30天，**切记需要每30天进行定期更换，或者每次请求都拉取新token**；

例如此接口，使用HTTPS POST发送：

https://aip.baidubce.com/rest/2.0/face/v2/identify?access\_token=24.f9ba9c5341b67688ab4added8bc91dec.2592000.1485570332.282335-8574074

POST中Body的参数，按照下方请求参数说明选择即可。

# 19.语音校验码接口

## 19.1.接口描述

此接口主要用于生成随机码，用于视频的语音识别校验使用，以判断视频的即时性，而非事先录制的，提升作弊的难度。

## 19.2.请求说明

**请求示例**

HTTP方法：POST

请求URL： https://aip.baidubce.com/rest/2.0/face/v1/faceliveness/sessioncode

URL参数：

| 参数 | 值 |
| --- | --- |
| access\_token | 通过API Key和Secret Key获取的access\_token,参考“[Access Token获取](http://ai.baidu.com/docs#Beginner-Auth)” |

Header：

| 参数 | 值 |
| --- | --- |
| Content-Type | application/x-www-form-urlencoded |

Body中放置请求参数，参数详情如下：

**请求参数**

| 参数 | 必选 | 类型 | 说明 |
| --- | --- | --- | --- |
| appid | 否 | string | 百度云创建应用时的唯一标识ID |

## 19.3.返回说明

**返回参数**

| 字段 | 必选 | 类型 | 说明 |
| --- | --- | --- | --- |
| session\_id | 是 | string | 语音校验码会话id |
| code | 是 | string | 语音验证码，数字形式，3~6位数字 |

**返回示例**

{

"err\_no": 0,

"err\_msg": "SUCCESS",

"result": {

"session\_id": "S59faeeebb9111890355690",

"code": "9940"

},

"timestamp": 1509617387,

"cached": 0,

"serverlogid": "0587756642"

}

# 20.视频活体检测接口

## 20.1.接口描述

此接口一方面通过语音识别得到校验码，通过session code来判断视频是否作弊。另一方面进行视频抽帧，判断是否为活体。

## 20.2.请求说明

**请求示例**

HTTP方法：POST

请求URL： https://aip.baidubce.com/rest/2.0/face/v1/faceliveness/verify

URL参数：

| 参数 | 值 |
| --- | --- |
| access\_token | 通过API Key和Secret Key获取的access\_token,参考“[Access Token获取](http://ai.baidu.com/docs#Beginner-Auth)” |

Header：

| 参数 | 值 |
| --- | --- |
| Content-Type | application/x-www-form-urlencoded |

Body中放置请求参数，参数详情如下：

**请求参数**

| 参数 | 必选 | 类型 | 说明 |
| --- | --- | --- | --- |
| session\_id | 是 | string | 语音校验码会话id，使用此接口的前提是已经调用了语音校验码接口 |
| video\_base64 | 是 | string | base64编码后的视频数据（视频限制：最佳为上传5-15s的mp4文件。视频编码方式：h264编码；音频编码格式：aac，pcm均可。） |

## 20.3.返回说明

**返回参数**

| 字段 | 必选 | 类型 | 说明 |
| --- | --- | --- | --- |
| score | 是 | float | 活体检测分数 |
| thresholds | 是 | array | 阈值参考，实际业务应用中，请以score>阈值判定通过，可直接选择不同误识别率的阈值，无需对应具体的分值，选择阈值参数即可。 |
| code | 是 | array | 语音校验码信息 |
| create | 是 | string | 生成的校验码，通过create和identify两个字段的对比，可以判断上传的视频是否为目标视频。 |
| identify | 是 | string | 语音识别出来的校验码 |
| pic\_list | 是 | array | 抽取图片信息列表 |
| pic\_list[i].face\_id | 是 | string | face唯一ID |
| pic\_list[i].pic | 是 | string/encryption | base64编码后的图片信息 |

**返回示例**

{

err\_no:0,

err\_msg: 'success',

result: {

score: 0.984654366,

thresholds: {

"frr\_1e-4": 0.05, //万分之一误识别率的阈值

"frr\_1e-3": 0.3, //千分之一误识别率的阈值

"frr\_1e-2": 0.9 //百分之一误识别率的阈值

},

code: {

"create": "5789",

"identify": "5789"

},

pic\_list: [

{

"face\_id": 5745745747,

"pic": "gsagaheryzxv..."

},

{

"face\_id": 5745745747,

"pic": "gsagaheryzxv..."

}

]

},

"timestamp": 1509611848,

"cached": 0,

"serverlogid": "2248375729"

}

错误码列表

| 错误码 | 错误信息 | 描述 |
| --- | --- | --- |
| 216506 | redis操作失败 |  |
| 216505 | redis连接失败 |  |
| 216430 | rtse/face 服务异常 |  |
| 216502 | 当前会话已失效 |  |
| 216501 | 没有找到人脸 |  |
| 216508 | 没有找到视频信息 |  |
| 216434 | 活体检测失败 |  |
| 216908 | 视频中人脸质量过低(返回的错误信息 会包含具体的错误信息包含 illumiantion(光照不足)angle(角度) blur(人脸模糊) occlusion(遮挡)too large(人脸过大) 等原因 |  |
| 216507 | 视频中有多张人脸 |  |
| 216433 | 视频解析服务发生错误 |  |
| 216432 | 视频解析服务调用失败 |  |
| 216431 | 语音识别服务异常 |  |
| 216500 | 验证码位数错误 |  |

# 21.错误码

若请求错误，服务器将返回的JSON文本包含以下参数：

* **error\_code**：错误码。
* **error\_msg**：错误描述信息，帮助理解和解决发生的错误。

例如Access Token失效返回：

{

"error\_code": 110,

"error\_msg": "Access token invalid or no longer valid"

}

上述问题重新获取新的Access Token再次请求即可。

## 21.1.接口流控及鉴权错误码

| 错误码 | 错误信息 | 描述 | 处理建议 |
| --- | --- | --- | --- |
| 4 | Open api request limit reached | 集群超限额 | 可直接[提交工单](http://ticket.bce.baidu.com/#/ticket/create)，咨询问题类型请选择**人工智能服务**； |
| 6 | no permission to access data | 没有接口权限 | 请确认您调用的接口已经被赋权 常见问题是有V3版本权限， 调用的是v2版本接口； |
| 17 | Open api daily request limit reached | 每天流量超限额 | 可[提交工单](http://ticket.bce.baidu.com/#/ticket/create)，咨询问题类型请选择**人工智能服务**； |
| 18 | Open api qps request limit reached | QPS超限额 | 可直接自助[购买更多QPS](http://ai.baidu.com/docs#/Face-Pricing/top)、联系商务接口人、或者[提交工单](http://ticket.bce.baidu.com/#/ticket/create)，咨询问题类型请选择**人工智能服务**； |
| 19 | Open api total request limit reached | 请求总量超限额 | 可[提交工单](http://ticket.bce.baidu.com/#/ticket/create)，咨询问题类型请选择**人工智能服务**； |
| 100 | Invalid parameter | 无效的access\_token参数 | token拉取失败，可以参考“[Access Token获取](http://ai.baidu.com/docs#Beginner-Auth)”重新获取 |
| 110 | Access token invalid or no longer valid | Access Token失效 | **token有效期为30天，注意需要定期更换**，也可以每次请求都拉取新token |
| 111 | Access token expired | Access token过期 | **token有效期为30天，注意需要定期更换**，也可以每次请求都拉取新token |

## 21.2.通用及业务错误码

| **错误码** | **错误信息** | **说明** | **处理建议** |
| --- | --- | --- | --- |
| 222001 | param[] is null | 必要参数未传入 | 参考API说明文档，修改参数 |
| 222002 | param[start] format error | 参数格式错误 | 参考API说明文档，修改参数 |
| 222003 | param[length] format error | 参数格式错误 | 参考API说明文档，修改参数 |
| 222004 | param[op\_app\_id\_list] format error | 参数格式错误 | 参考API说明文档，修改参数 |
| 222005 | param[group\_id\_list] format error | 参数格式错误 | 参考API说明文档，修改参数 |
| 222006 | group\_id format error | 参数格式错误 | 参考API说明文档，修改参数 |
| 222007 | uid format error | 参数格式错误 | 参考API说明文档，修改参数 |
| 222008 | face\_id format error | 参数格式错误 | 参考API说明文档，修改参数 |
| 222009 | quality\_conf format error | 参数格式错误 | 参考API说明文档，修改参数 |
| 222010 | user\_info format error | 参数格式错误 | 参考API说明文档，修改参数 |
| 222011 | param[uid\_list] format error | 参数格式错误 | 参考API说明文档，修改参数 |
| 222012 | param[op\_app\_id] format error | 参数格式错误 | 参考API说明文档，修改参数 |
| 222013 | param[image] format error | 参数格式错误 | 参考API说明文档，修改参数 |
| 222014 | param[app\_id] format error | 参数格式错误 | 参考API说明文档，修改参数 |
| 222015 | param[image\_type] format error | 参数格式错误 | 参考API说明文档，修改参数 |
| 222016 | param[max\_face\_num] format error | 参数格式错误 | 参考API说明文档，修改参数 |
| 222017 | param[face\_field] format error | 参数格式错误 | 参考API说明文档，修改参数 |
| 222018 | param[user\_id] format error | 参数格式错误 | 参考API说明文档，修改参数 |
| 222019 | param[quality\_control] format error | 参数格式错误 | 参考API说明文档，修改参数 |
| 222020 | param[liveness\_control] format error | 参数格式错误 | 参考API说明文档，修改参数 |
| 222021 | param[max\_user\_num] format error | 参数格式错误 | 参考API说明文档，修改参数 |
| 222022 | param[id\_card\_number] format error | 参数格式错误 | 参考API说明文档，修改参数 |
| 222023 | param[name] format error | 参数格式错误 | 参考API说明文档，修改参数 |
| 222024 | param[face\_type] format error | 参数格式错误 | 参考API说明文档，修改参数 |
| 222025 | param[face\_token] format error | 参数格式错误 | 参考API说明文档，修改参数 |
| 222026 | param[max\_star\_num] format error | 参数格式错误 | 参考API说明文档，修改参数 |
| 222201 | network not available | 服务端请求失败 | 重新尝试 |
| 222202 | pic not has face | 图片中没有人脸 | 检查图片质量 |
| 222203 | image check fail | 无法解析人脸 | 检查图片质量 |
| 222204 | image\_url\_download\_fail | 从图片的url下载 图片失败 | 请确认url可公网访问 |
| 222205 | network not availablel | 服务端请求失败 | 重新尝试 |
| 222206 | rtse service return fail | 服务端请求失败 | 重新尝试 |
| 222207 | match user is not found | 未找到匹配的用户 |  |
| 222208 | the number of image is incorrect | 图片的数量错误 | 多张图片请使用json格式传输 |
| 222209 | face token not exist | face token不存在 | 检查face\_token |
| 222300 | add face fail | 人脸图片添加失败 | 重新尝试 |
| 222301 | get face fail | 获取人脸图片失败 |  |
| 222302 | system error | 服务端请求失败 | 重新尝试 |
| 222303 | get face fail | 获取人脸图片失败 |  |
| 223100 | group is not exist | 操作的用户组不存在 |  |
| 223101 | group is already exist | 该用户组已存在 |  |
| 223102 | user is already exist | 该用户已存在 |  |
| 223103 | user is not exist | 找不到该用户 |  |
| 223104 | group\_list is too large | group\_list包含组 数量过多 |  |
| 223105 | face is already exist | 该人脸已存在 |  |
| 223106 | face is not exist | 该人脸不存在 |  |
| 223110 | uid\_list is too large | uid\_list包含数量过多 |  |
| 223111 | dst group is not exist | 目标用户组不存在 |  |
| 223112 | quality\_conf format error | quality\_conf格式不正确 |  |
| 223113 | face is covered | 人脸有被遮挡 |  |
| 223114 | face is fuzzy | 人脸模糊 |  |
| 223115 | face light is not good | 人脸光照不好 |  |
| 223116 | incomplete face | 人脸不完整 |  |
| 223117 | app\_list is too large | app\_list包含app数量 过多 |  |
| 223118 | quality control error | 质量控制项错误 |  |
| 223119 | liveness control item error | 活体控制项错误 |  |
| 223120 | liveness check fail | 活体检测未通过 |  |
| 223121 | left eye is occlusion | 质量检测未通过 左眼 遮挡程度过高 |  |
| 223122 | right eye is occlusion | 质量检测未通过 右眼 遮挡程度过高 |  |
| 223123 | left cheek is occlusion | 质量检测未通过 左脸 遮挡程度过高 |  |
| 223124 | right cheek is occlusion | 质量检测未通过 右脸 遮挡程度过高 |  |
| 223125 | chin contour is occlusion | 质量检测未通过 下巴遮挡程度过高 |  |
| 223126 | nose is occlusion | 质量检测未通过 鼻子遮挡程度过高 |  |
| 223127 | mouth is occlusion | 质量检测未通过 嘴巴 遮挡程度过高 |  |
| 222901 | system busy | 系统繁忙 | 请重新尝试 |
| 222902 | system busy | 系统繁忙 | 请重新尝试 |
| 222903 | system busy | 系统繁忙 | 请重新尝试 |
| 222904 | system busy | 系统繁忙 | 请重新尝试 |
| 222905 | system busy | 系统繁忙 | 请重新尝试 |
| 222906 | system busy | 系统繁忙 | 请重新尝试 |
| 222907 | system busy | 系统繁忙 | 请重新尝试 |
| 222908 | system busy | 系统繁忙 | 请重新尝试 |
| 222909 | system busy | 系统繁忙 | 请重新尝试 |
| 222910 | system busy | 系统繁忙 | 请重新尝试 |
| 222911 | system busy | 系统繁忙 | 请重新尝试 |
| 222912 | system busy | 系统繁忙 | 请重新尝试 |
| 222913 | system busy | 系统繁忙 | 请重新尝试 |
| 222914 | system busy | 系统繁忙 | 请重新尝试 |
| 222915 | system busy | 系统繁忙 | 请重新尝试 |
| 222916 | system busy | 系统繁忙 | 请重新尝试 |
| 222361 | system busy | 系统繁忙 | 请重新尝试 |

## 21.3.公安验证接口错误码

| 错误码 | 错误信息 | 说明 | 处理建议 |
| --- | --- | --- | --- |
| 222350 | police picture is none or low quality | 公安网图片不存在或 质量过低 |  |
| 222351 | id number and name not match or id number not exist | 身份证号与姓名不匹配或该 身份证号不存在 |  |
| 222352 | name format error | 身份证名字格式错误 |  |
| 222353 | id number format error | 身份证号码格式错误 |  |
| 222354 | id number not exist | 公安库里不存在此身份证号 |  |
| 222355 | police picture not exist | 身份证号码正确，公安库里没有 对应的照片 |  |
| 222360 | invalid name or id number | 身份证号码或名字非法（公安网校 验不通过） |  |