人脸识别

SDK 文档

产品文档

腾讯云

文档目录

SDK 文档
SDK 概览

最近更新时间：2018-08-03 17:12:37

除了直接使用 API 接口外，腾讯云智能图像识别还提供了丰富多样的 SDK 供开发者使用。

图像识别 SDK

SDK	接入文档
Java-SDK-图像识别	Java-SDK- 图像识别
Python-SDK-图像识别	Python-SDK- 图像识别
CPP-SDK-图像识别	CPP-SDK- 图像识别
PHP-SDK-图像识别	PHP-SDK- 图像识别
Node-SDK-图像识别	Node-SDK-图像识别

Java-SDK-图像识别

最近更新时间：2018-08-08 19:43:12

SDK 获取

智能图像的 Java SDK 下载地址：Java-SDK-V2.0。

使用前准备

1. 前往注册： 腾讯云账号注册 （详细指引见 注册腾讯云 ）。
2. 取得 **APPID**、**SecretId**、**SecretKey**：请前往 云API密钥 ，单击“新建密钥”（目前只支持主账号及密钥进行调用）。

如何集成到您的项目中

获得 SDK jar 文件

1. 直接使用 release/*-with-dependencies.jar
2. 或者自行编译：在工程根目录下执行命令 mvn assembly:assembly ，编译结果见 target/*-with-dependencies.jar

导入 jar 到项目中

根据项目具体情况导入 *-with-dependencies.jar

使用简介

初始化

```
ImageClient imageClient = new ImageClient(APPID, SecretId, SecretKey);
```

设置代理

根据实际网络环境，可能要设置代理，例如：

```
Proxy proxy = new Proxy(Type.HTTP, new InetSocketAddress("127.0.0.1", 8080));
imageClient.setProxy(proxy);
```

使用

SDK 提供功能如下：

图像识别：鉴黄，标签

文字识别(OCR)：身份证，名片，通用，驾驶证行驶证，营业执照，银行卡，车牌号

人脸识别：人脸检测，五官定位，个体信息管理，人脸验证，人脸对比及人脸检索

人脸核身：照片核身（通过照片和身份证信息），获取唇语验证码（用于活体核身），活体核身（通过视频和照片），活体核身（通过视频和身份证信息）

```
// 调用车牌识别API示例
String imageUrl = "http://youtu.qq.com/app/img/experience/char_general/icon_ocr_license_3.jpg";
String result = imageClient.ocrPlate(new OcrPlateRequest("bucketName", imageUrl));
System.out.println(result);
```

更多例子详情可参见 Demo.java 的代码。

如何运行这个 Demo 工程

1. 修改文件 src/main/java/com/qcloud/image/demo/Demo.java 的 main() 方法，填入上述申请到的 **APPID**、**SecretId**、**SecretKey**

2. 导入到 IDE：这个 Demo 工程是用 Maven 构建的，以 Intellij IDEA 为例，导入方式为：Import Project -> 选择工程目录 ->
   Import project from external model -> Maven

3. 运行：Demo.java 右键，Run Demo.main()

Python-SDK-图像识别

最近更新时间：2018-08-08 19:44:26

开发准备

SDK 获取

智能图像的 Python SDK 下载地址：Python-SDK-V2.0 。

开发准备

使用pip

Python 2:

pip install qcloud_image

Python 3:

pip3 install qcloud_image

快速入门

在腾讯云申请业务的授权

授权包括： APPID 、SecretId 、 SecretKey ，目前只支持主账号及密钥进行调用。

> **注意**: BUCKET 为历史遗留字段, 无需修改。

创建对应操作类的对象

如果要使用图片，需要创建图片操作类对象：

```
from qcloud_image import Client
from qcloud_image import CIUrl, CIFile, CIBuffer, CIUrls, CIFiles, CIBuffers
appid = 'APP_ID'
secret_id = 'SECRET_ID'
secret_key = 'SECRET_KEY'
bucket = 'BUCKET'
client = Client(appid, secret_id, secret_key, bucket)
client.use_http()
client.set_timeout(30)
```

调用对应的方法

在创建完对象后，根据实际需求，调用对应的操作方法就可以了。sdk 提供的方法包括：图片识别、人脸识别及人脸核身等。

图片识别

图片识别包括：图片鉴黄、图片标签、OCR - 身份证识别及 OCR - 名片识别。

图片鉴黄

```
//单个或多个图片Url
print (client.porn_detect(CIUrls(['http://jiangsu.china.com.cn/uploadfile/2015/1102/1446443026382534.jpg','http://n.sinaimg.cn/fashion/transform/20160704/flgG-fxtspsa6612705.jpg'])))
//单个或多个图片File
print (client.porn_detect(CIFiles(['./test.jpg',])))
```

图片标签

```
//单个图片url
print (client.tag_detect(CIUrl('http://img3.a0bi.com/upload/ttq/20160814/1471155260063.png')))
//单个图片file
print (client.tag_detect(CIFile('./hot2.jpg')))
```

OCR - 身份证识别

```
//单个或多个图片Url,识别身份证正面
print (client.idcard_detect(CIUrls(['http://imgs.focus.cn/upload/sz/5876/a_58758051.jpg']), 0))
//单个或多个图片file,识别身份证正面
print (client.idcard_detect(CIFiles(['./id4zheng.jpg','./id1zheng.jpg']), 0))
//单个或多个图片Url,识别身份证反面
print (client.idcard_detect(CIUrls(['http://www.csx.gov.cn/cwfw/bszn/201403/W020121030349825312574.jpg', 'http://www.4009951551.com/upload/image/20151026/1445831136187479.png']), 1))
//单个或多个图片file,识别身份证反面
print (client.idcard_detect(CIFiles(['./id5_fan.jpg']), 1))
```

OCR - 名片识别

```
//单个或多个图片Url
print (client.namecard_detect(CIUrls(['http://pic1.nipic.com/2008-12-03/2008123181119306_2.jpg', 'http://pic.58pic.com/58pic/12/49/04/80k58PICzYP.jpg'])))
//单个或多个图片file
print (client.namecard_detect(CIFiles(['./name1.jpg'])))
```

人脸识别

人脸识别包括：人脸检测、五官定位、个体信息管理、人脸验证、人脸对比及人脸检索。

人脸检测

```
//单个图片Url, mode:1为检测最大的人脸，0为检测所有人脸
print (client.face_detect(CIUrl('http://img3.a0bi.com/upload/ttq/20160814/1471155260063.png')))
//单个图片file,mode:1为检测最大的人脸，0为检测所有人脸
print (client.face_detect(CIFile('./hot2.jpg')))
```

五官定位

```
//单个图片Url,mode:1为检测最大的人脸，0为检测所有人脸
print (client.face_shape(CIUrl('http://img3.a0bi.com/upload/ttq/20160814/1471155260063.png'),1))
//单个图片file,mode:1为检测最大的人脸，0为检测所有人脸
print (client.face_shape(CIFile('./hot2.jpg'),1))
```

个体信息管理

```
//个体创建,创建一个Person，并将Person放置到group_ids指定的组当中，不存在的group_id会自动创建。
//创建一个Person, 使用图片url
print (client.face_newperson('person111', ['group2',], CIUrl('http://img3.a0bi.com/upload/ttq/20160814/1471155260063.png'), 'xiaoxin'))
//创建一个Person, 使用图片file
print (client.face_newperson('person211', ['group2',], CIFile('./hot2.jpg')))
//增加人脸,将一组Face加入到一个Person中。注意，一个Face只能被加入到一个Person中。
//将单个或者多个Face的url加入到一个Person中
print (client.face_addface('person111', CIUrls(['http://jiangsu.china.com.cn/uploadfile/2015/1102/1446443026382534.jpg','http://n.sinaimg.cn/fashion/transform/20160704/flgG-fxtspsa6612705.jpg'])))
//将单个或者多个Face的file加入到一个Person中
print (client.face_addface('person211', CIFiles(['./test.jpg',])))
//删除人脸,删除一个person下的face
print (client.face_delface('person111', ['person111',]))
//设置信息
print (client.face_setinfo('person111', 'hello'))
//获取信息
print (client.face_getinfo('person111'))
//获取组列表
print (client.face_getgroupids())
//获取人列表
print (client.face_getpersonids('group2'))
//获取人脸列表
print (client.face_getfaceids('person211'))
```

```
//获取人脸信息
print (client.face_getfaceinfo('1820307972625034938'))
//删除个人
print (client.face_delperson('person11'))
```

人脸验证

给定一个 Face 和一个 Person ，返回是否是同一个人的判断以及置信度。

```
//人脸验证,单个图片Url
print (client.face_verify('person111', CIUrl('http://img3.a0bi.com/upload/ttq/20160814/1471155260063.png')))
//人脸验证,单个图片file
print (client.face_verify('person111', CIFile('./test.jpg')))
```

人脸对比

```
//两个对比图片的文件url
print (client.face_compare(CIFile('./zhao1.jpg'), CIFile('./zhao2.jpg')))
//两个对比图片的文件file
print (client.face_compare(CIUrl('http://www.miexue.com/d/file/junshiyingshi/2016-12-05/60bce03aac7a57e4fc600ecee1591e1d.jpg'), CIUrl('http://img.mp.itc.cn/upload/20161118/ee6be67ec6fb4135b5d579ab05acd715_th.jpg')))
//一个是图片的文件url， 一个是对比图片的文件file
print (client.face_compare(CIFile('./zhao1.jpg'), CIUrl('http://www.miexue.com/d/file/junshiyingshi/2016-12-05/60bce03aac7a57e4fc600ecee1591e1d.jpg')))
```

人脸检索

对于一个待识别的人脸图片，在一个 Group 中识别出最相似的 Top5 Person 作为其身份返回，返回的Top5中按照相似度从大到小排列。

```
//人脸检索,单个文件url
print (client.face_identify('group1', CIUrl('http://www.5djiaren.com/uploads/2016-07/22-141354_227.jpg')))
//人脸检索,单个文件file
print (client.face_identify('group2', CIFile('./test.jpg')))
```

人脸核身

身份证识别对比

```
#身份证url
print (client.face_idcardcompare('420822198804266119', '李时杰', CIUrl('http://docs.ebdoor.com/Image/CompanyCertificate/1/16844.jpg')))
#身份证文件file
print (client.face_idcardcompare('420822198804266119', '李时杰', CIFile('./id4_zheng.jpg')))
```

活体检测 - 获取唇语验证码

```
obj = client.face_livegetfour()
print (obj)
#验证码
validate_data = obj['data']['validate_data']
```

活体检测 - 视频身份信息核验

```
print (client.face_livedetectfour(validate_data, CIFile('../dn.qlv'), False, CIFile('../wxb.jpg')))
```

活体检测 - 视频与用户照片的比对

```
print (client.face_idcardlivedetectfour(validate_data, CIFile('../dnn.qlv'), '330782198802084329', '李时杰'))
```

CPP-SDK-图像识别

最近更新时间：2018-08-08 19:44:42

开发准备

> **注意：**以下为 Linux 等类 UINX 系统使用手册，暂不支持 Windows 系统。

SDK 获取

智能图像 C++ SDK 下载地址：cpp-SDK-V2.0 。

开发准备

依赖静态库: curl jsoncpp (在 lib 文件夹下)；
依赖动态库: ssl crypto rtz (需要安装)；
(1)安装 openssl 的库和头文件 http://www.openssl.org/source/ ；
(2)安装 curl 的库和头文件 http://curl.haxx.se/download/curl-7.43.0.tar.gz ；
(3)安装 jsoncpp 的库和头文件 https://github.com/open-source-parsers/jsoncpp ；
(4)安装 cmake 工具 http://www.cmake.org/download/ 。

SDK 配置

直接下载 github 上提供的源代码，集成到您的开发环境。

执行下面的命令：

```
cd ${image-cpp-sdk-v2.0}
mkdir -p build
cd build
cmake ..
make
```

image_demo.cpp 里面有常见 API 的例子。生成的 image_demo 可直接运行，生成的静态库名称为：libimagesdk.a 。生成的 libimagesdk.a 放到用户自己的工程里 lib 路径下，include 目录拷贝到用户的工程的 include 路径下。

快速入门

在腾讯云申请业务的授权

授权包括： APPID 、SecretId 、 SecretKey ，目前只支持主账号及密钥进行调用。

创建对应操作类的对象

如果要使用图片，需要创建图片操作类对象：

```
//设置全局参数（非必须）
ImageSysConfig::setAuthExpiredTime(300); //设置签名超时时长300s
//生成ImageAPI对象
ImageConfig config(APP_ID, SECRET_ID, SECRET_KEY);
ImageAPI image(config);
```

调用对应的方法

在创建完对象后，根据实际需求，调用对应的操作方法就可以了。SDK 提供的方法包括：图片识别、人脸识别及人脸核身等。

图片识别

图片识别包括：图片鉴黄、图片标签、OCR - 身份证识别及 OCR - 名片识别。

图片鉴黄

```
//单个或多个图片Url
vector<string> pornUrls; pornUrls.push_back("http://hearthstone.nos.netease.com/1/artworkGvG/GoblinBlastmagel.jpg");
pornUrls.push_back("http://hearthstone.nos.netease.com/1/artworknaxx/Faerlinal.jpg");
pornUrls.push_back("http://hearthstone.nos.netease.com/1/artworknaxx/KelThuzadl.jpg");
PornDetectReq pornReq(BUCKET, pornUrls);
ret = image.PornDetect(pornReq);
cout<<ret<<endl;
//单个或多个图片File
map<string, string> pornImages;
pornImages["1.jpg"] = FileUtil::getFileContent("pic/1.jpg");
pornImages["2.jpg"] = FileUtil::getFileContent("pic/2.jpg");
pornImages["3.jpg"] = FileUtil::getFileContent("pic/3.jpg");
PornDetectReq pornReq2(BUCKET, pornImages);
ret = image.PornDetect(pornReq);
cout<<ret<<endl;
```

图片标签

```
//单个图片url
TagDetectReq tagReq(BUCKET);
tagReq.SetUrl("http://img3.a0bi.com/upload/ttq/20160814/1471155260063.png");
```

```
ret = image.TagDetect(tagReq);
cout<<ret<<endl;
//单个图片file
TagDetectReq tagReq(BUCKET);
tagReq.SetImage("hot1.jpg");
ret = image.TagDetect(tagReq);
cout<<ret<<endl;
```

OCR - 身份证识别

```
//单个或多个图片Url,识别身份证正面
vector<string> idZUrls;
idZUrls.push_back("http://imgs.focus.cn/upload/sz/5876/a_58758051.jpg");
idZUrls.push_back("http://img5.iqilu.com/c/u/2013/0530/1369896921237.jpg");
IdCardOcrReq idReq(BUCKET, idZUrls,0);
ret = image.IdCardOcr(idReq);
cout<<ret<<endl;
//单个或多个图片file,识别身份证正面
map<string, string> idZImages;
idZImages["id6zheng.jpg"] = FileUtil::getFileContent("id6zheng.jpg");
idZImages["id2zheng.jpg"] = FileUtil::getFileContent("id2zheng.jpg");
IdCardOcrReq idReq2(BUCKET, idZImages, 0);
ret = image.IdCardOcr(idReq2);
cout<<ret<<endl;
//单个或多个图片Url,识别身份证反面
vector<string> idFUrls; idFUrls.push_back("http://www.csx.gov.cn/cwfw/bszn/201403/W020121030349
825312574.jpg"); idFUrls.push_back("http://www.4009951551.com/upload/image/20151026/14458311
36187479.png");
IdCardOcrReq idReq3(BUCKET, idFUrls,1);
ret = image.IdCardOcr(idReq3);
cout<<ret<<endl;
//单个或多个图片file,识别身份证反面
map<string, string> idFImages;
idFImages["id5fan.jpg"] = FileUtil::getFileContent("id5fan.jpg");
idFImages["id7fan.jpg"] = FileUtil::getFileContent("id7fan.jpg");
IdCardOcrReq idReq4(BUCKET, idFImages, 1);
ret = image.IdCardOcr(idReq4);
cout<<ret<<endl;
```

OCR - 名片识别

```
//单个或多个图片Url
vector<string> nameUrls;
nameUrls.push_back("http://pic1.nipic.com/2008-12-03/2008123181119306_2.jpg");
```

```
nameUrls.push_back("http://pic.58pic.com/58pic/12/49/04/80k58PICzYP.jpg");
NameCardOcrReq nameReq(BUCKET, nameUrls, 0);
ret = image.NameCardOcr(nameReq);
cout<<ret<<endl;
//单个或多个图片file
map<string, string> nameImages;
nameImages["r.jpg"] = FileUtil::getFileContent("r.jpg");
nameImages["name2.jpg"] = FileUtil::getFileContent("name2.jpg");
NameCardOcrReq nameReq2(BUCKET, nameImages, 0);
ret = image.NameCardOcr(nameReq2);
cout<<ret<<endl;
```

人脸识别

人脸识别包括：人脸检测、五官定位、个体信息管理、人脸验证、人脸对比及人脸检索。

人脸检测

```
//单个图片Url, mode:1为检测最大的人脸 , 0为检测所有人脸
FaceDetectReq faceDetectReq(BUCKET);
faceDetectReq.SetMode(0);
faceDetectReq.SetUrl("http://burningtest-10006599.cosgz.myqcloud.com/laobao.jpg");
ret = image.FaceDetect(faceDetectReq);
cout<<ret<<endl;
//单个图片file
faceDetectReq.SetImage("zhao2.jpg");
ret = image.FaceDetect(faceDetectReq);
cout<<ret<<endl;
```

五官定位

```
//单个图片Url,检测最大的人脸
FaceShapeReq faceShapeReq(BUCKET);
faceShapeReq.SetMode(0);
faceShapeReq.SetUrl("http://burningtest-10006599.cosgz.myqcloud.com/laobao.jpg");
ret = image.FaceShape(faceShapeReq);
cout<<ret<<endl;
//单个图片file
faceShapeReq.SetImage("zhao2.jpg");
ret = image.FaceShape(faceShapeReq);
cout<<ret<<endl;
```

个体信息管理

```
//创建一个Person，并将Person放置到group_ids指定的组当中, 使用图片url
FaceNewPersonReq newPersonReq(BUCKET);
newPersonReq.SetUrl("http://img3.a0bi.com/upload/ttq/20160814/1471155260063.png");
newPersonReq.SetPersonId("person2222");
newPersonReq.AddGroupId("group2222");
ret = image.FaceNewPerson(newPersonReq);
cout<<ret<<endl;
//单个图片file
newPersonReq.SetPersonId("person3333");
newPersonReq.SetImage("zhao2.jpg");
ret = image.FaceNewPerson(newPersonReq);
cout<<ret<<endl;
//增加人脸,将单个或者多个Face的url加入到一个Person中.
FaceAddFaceReq addFaceReq(BUCKET);
addFaceReq.AddUrl("http://jiangsu.china.com.cn/uploadfile/2015/1102/1446443026382534.jpg");
addFaceReq.AddUrl("http://n.sinaimg.cn/fashion/transform/20160704/flgG-fxtspsa6612705.jpg");
addFaceReq.SetPersonId("person2222");
ret = image.FaceAddFace(addFaceReq);
cout<<ret<<endl;
//增加人脸,将单个或者多个Face的file加入到一个Person中
addFaceReq.AddImage("zhao1.jpg");
addFaceReq.AddImage("zhao2.jpg");
addFaceReq1.SetPersonId("person2222");
ret = image.FaceAddFace(addFaceReq);
cout<<ret<<endl;
//删除人脸
FaceDelFaceReq delFaceReq(BUCKET);
delFaceReq.SetPersonId("person2222");
delFaceReq.AddFaceId("1831408218312574949");
delFaceReq.AddFaceId("1831408248150847230");
ret = image.FaceDelFace(delFaceReq);
cout<<ret<<endl;
//设置信息
FaceSetInfoReq setInfoReq(BUCKET);
setInfoReq.SetPersonId("person2222");
setInfoReq.SetPersonName("ying");
ret = image.FaceSetInfo(setInfoReq);
cout<<ret<<endl;
//获取信息
FaceGetInfoReq getInfoReq(BUCKET);
getInfoReq.SetPersonId("person2222");
ret = image.FaceGetInfo(getInfoReq);
cout<<ret<<endl;
//获取组列表
FaceGetGroupIdsReq getGroupIdReq(BUCKET);
```

```
ret = image.FaceGetGroupIds(getGroupIdReq);
cout<<ret<<endl;
//获取人列表
FaceGetPersonIdsReq getPersonIdReq(BUCKET);
getPersonIdReq.SetGroupId("group2222");
ret = image.FaceGetPersonIds(getPersonIdReq);
cout<<ret<<endl;
//获取人脸列表
FaceGetFaceIdsReq getFaceIdReq(BUCKET);
getFaceIdReq.SetPersonId("person2222");
ret = image.FaceGetFaceIds(getFaceIdReq);
cout<<ret<<endl;
//获取人脸信息
FaceGetFaceInfoReq getFaceInfoReq(BUCKET);
getFaceInfoReq.SetFaceId("1704147773393235686");
ret = image.FaceGetFaceInfo(getFaceInfoReq);
cout<<ret<<endl;
//删除个人
FaceDelPersonReq delPersonReq(BUCKET);
delPersonReq.SetPersonId("person2222");
ret = image.FaceDelPerson(delPersonReq);
cout<<ret<<endl;
```

人脸验证

给定一个 Face 和一个 Person ，返回是否是同一个人的判断以及置信度：

```
//单个图片Url
FaceVerifyReq faceVerifyReq(BUCKET);
faceVerifyReq.SetUrl("http://img3.a0bi.com/upload/ttq/20160814/1471155260063.png");
faceVerifyReq.SetPersonId("person1");
ret = image.FaceVerify(faceVerifyReq);
cout<<ret<<endl;
//单个图片file
faceVerifyReq.SetImage("yang3.jpg");
ret = image.FaceVerify(faceVerifyReq);
cout<<ret<<endl;
```

人脸对比

```
//两个对比图片的文件url
FaceCompareReq fcReq(BUCKET);
fcReq.AddUrl("http://burningtest-10006599.cosgz.myqcloud.com/laobao.jpg");
fcReq.AddUrl("http://burningtest-10006599.cosgz.myqcloud.com/laobao.jpg");
```

```
ret = image.FaceCompare(fcReq);
cout<<ret<<endl;
//两个对比图片的文件file
fcReq.AddImage("zhao1.jpg");
fcReq.AddImage("zhao2.jpg");
ret = image.FaceCompare(fcReq);
cout<<ret<<endl;
```

人脸检索

对一张待识别的人脸图片，在一个或多个 group 中识别出最相似的 Top5 person 作为其身份返回，返回的 Top5 中按照相似度从大到小排列。

```
//单个图片Url
FaceIdentifyReq identifyReq(BUCKET);
identifyReq.SetUrl("http://img3.a0bi.com/upload/ttq/20160814/1471155260063.png");
identifyReq.SetGroupId("group2222");
ret = image.FaceIdentify(identifyReq);
cout<<ret<<endl;
//单个图片file
identifyReq.SetImage("yang3.jpg");
ret = image.FaceIdentify(identifyReq);
cout<<ret<<endl;
```

人脸核身

身份证识别对比

```
FaceIdCardCompareReq idCompareReq(BUCKET);
idCompareReq.SetUrl("http://docs.ebdoor.com/Image/CompanyCertificate/1/16844.jpg");
idCompareReq.SetIdCardNumber("330782198802084329");
idCompareReq.SetIdCardName("季锦锦");
ret = image.FaceIdCardCompare(idCompareReq);
idCompareReq.SetImage("idcard.jpg");
ret = image.FaceIdCardCompare(idCompareReq);
cout<<ret<<endl;
```

活体检测 - 获取唇语验证码

```
FaceLiveGetFourReq getFourReq(BUCKET);
ret = image.FaceLiveGetFour(getFourReq);
cout<<ret<<endl;
```

```
string validate = "";
Json::Value obj = StringUtil::StringToJson(ret);
```

活体检测 - 视频与用户照片的比对

```
FaceLiveDetectFourReq detectFourReq(BUCKET);
detectFourReq.SetValidateData(validate);
detectFourReq.SetVideo("ZOE_0171.mp4");
ret = image.FaceLiveDetectFour(detectFourReq);
cout<<ret<<endl;
```

活体检测 - 视频身份信息核验

```
FaceIdCardLiveDetectFourReq iddetectFourReq(BUCKET);
iddetectFourReq.SetValidateData(validate);
iddetectFourReq.SetVideo("ZOE_0171.mp4");
iddetectFourReq.SetIdCardName("季锦锦");
iddetectFourReq.SetIdCardNumber("330782198802084329");
ret = image.FaceIdCardLiveDetectFour(iddetectFourReq);
cout<<ret<<endl;
return 0;
```

PHP-SDK-图像识别

最近更新时间：2018-10-24 20:24:23

开发准备

SDK 获取

智能图像的 PHP SDK 下载地址：PHP-SDK-V2.0。

快速入门

在腾讯云申请业务的授权

授权包括： APPID 、SecretId 、SecretKey ，目前只支持主账号及密钥进行调用。

> **注意：**BUCKET 为历史遗留字段，无需修改。

创建对应操作类的对象

如果要使用图片，需要创建图片操作类对象：

```
require_once DIR . '/index.php';
use QcloudImage\CIClient;
$client = new CIClient('APP_ID', 'SECRET_ID', 'SECRET_KEY', 'BUCKET');
$client->setTimeout(30);
```

调用对应的方法

在创建完对象后，根据实际需求，调用对应的操作方法就可以了。SDK 提供的方法包括：图片识别、人脸识别及人脸核身等。

图片识别

图片识别包括：图片鉴黄、图片标签、OCR - 身份证识别及 OCR - 名片识别。

图片鉴黄

```
//单个或多个图片 URL
var_dump ($client->pornDetect(array('urls'=>array('http://img3.a0bi.com/upload/ttq/20160814/147
```

```
1155260063.png',
"http://jiangsu.china.com.cn/uploadfile/2015/1102/1446443026382534.jpg"))));
//单个或多个图片 file
var_dump ($client->pornDetect(array('files'=>array('F:\pic\你好.jpg','G:\pic\test2.jpg'))));
```

图片标签

```
//单个图片 URL
var_dump ($client->tagDetect(array('url'=>'http://img3.a0bi.com/upload/ttq/20160814/1471155260
063.png')));
//单个图片 file
var_dump ($client->tagDetect(array('file'=>'G:\pic\hot1.jpg')));
//单个图片内容
var_dump ($client->tagDetect(array('buffer'=>file_get_contents('G:\pic\hot1.jpg'))));
```

OCR - 身份证识别

```
//单个或多个图片 URL，识别身份证正面
var_dump ($client->idcardDetect(array('urls'=>array('http://imgs.focus.cn/upload/sz/5876/a_587580
51.jpg', 'http://img5.iqilu.com/c/u/2013/0530/1369896921237.jpg')), 0));
//单个或多个图片 file，识别身份证正面
var_dump ($client->idcardDetect(array('files'=>array('F:\pic\id6zheng.jpg', 'F:\pic\id2zheng.jpg')), 0
));
//单个或多个图片内容，识别身份证正面
var_dump ($client->idcardDetect(array('buffers'=>array(file_get_contents('F:\pic\id6_zheng.jpg'),
file_get_contents('F:\pic\id2_zheng.jpg'))), 0));
//单个或多个图片 URL，识别身份证反面
var_dump ($client->idcardDetect(array('urls'=>array('http://www.csx.gov.cn/cwfw/bszn/201403/W02
0121030349825312574.jpg',
'http://www.4009951551.com/upload/image/20151026/1445831136187479.png')), 1));
//单个或多个图片 file，识别身份证反面
var_dump ($client->idcardDetect(array('files'=>array('F:\pic\id5fan.jpg', 'F:\pic\id7fan.png')), 1));
//单个或多个图片内容，识别身份证反面
var_dump ($client->idcardDetect(array('buffers'=>array(file_get_contents('F:\pic\id5_fan.jpg'),
file_get_contents('F:\pic\id7_fan.jpg'))), 1));
```

OCR - 名片识别

```
//单个或多个图片 HRL
var_dump ($client->namecardV2Detect(array('urls'=>array('http://open.youtu.qq.com/app/img/exp
erience/char_general/ocr_namecard_01.jpg'))));
//单个或多个图片 file
var_dump ($client->namecardV2Detect(array('files'=>array('assets/ocr_namecard_01.jpg'))));
```

```
//单个或多个图片内容
var_dump ($client->namecardV2Detect(array('buffers'=>array(file_get_contents('assets/ocr_namecard_01.jpg')))));
```

人脸识别

人脸识别包括：人脸检测、五官定位、个体信息管理、人脸验证、人脸对比及人脸检索。

人脸检测

```
//单个图片 URL， mode:1 为检测最大的人脸，0 为检测所有人脸
var_dump ($client->faceDetect(array('url'=>'http://img3.a0bi.com/upload/ttq/20160814/1471155260063.png'), 1));
//单个图片 file,mode:1 为检测最大的人脸，0 为检测所有人脸
var_dump ($client->faceDetect(array('file'=>'F:\pic\face1.jpg'),0));
//单个图片内容，mode:1 为检测最大的人脸，0 为检测所有人脸
var_dump ($client->faceDetect(array('buffer'=>file_get_contents('F:\pic\face1.jpg')), 1));
```

五官定位

```
//单个图片 URL，mode:1 为检测最大的人脸，0 为检测所有人脸
var_dump ($client->faceShape(array('url'=>'http://img3.a0bi.com/upload/ttq/20160814/1471155260063.png'),1));
//单个图片 file，mode:1 为检测最大的人脸，0 为检测所有人脸
var_dump ($client->faceShape(array('file'=>'F:\pic\face1.jpg'),0));
//单个图片内容，mode:1 为检测最大的人脸，0 为检测所有人脸
var_dump ($client->faceShape(array('buffer'=>file_get_contents('F:\pic\face1.jpg')), 1));
```

个体信息管理

```
//个体创建，创建一个 Person，并将 Person 放置到 group_ids 指定的组当中，不存在的 group_id 会自动创建。
//创建一个 Person，使用图片 URL
var_dump ($client->faceNewPerson('person1111', array('group11',), array('url'=>'http://img3.a0bi.com/upload/ttq/20160814/1471155260063.png'), 'xiaoxin'));
//创建一个 Person，使用图片 file
var_dump ($client->faceNewPerson('person2111', array('group11',), array('file'=>'F:\pic\hot1.jpg')));
//创建一个 Person，使用图片内容
var_dump ($client->faceNewPerson('person3111', array('group11',), array('buffer'=>file_get_contents('F:\pic\zhao1.jpg'))));
//增加人脸，将一组 Face 加入到一个 Person 中。
//将单个或者多个 Face 的 URL 加入到一个 Person 中
var_dump ($client->faceAddFace('person1111', array('urls'=>array('http://jiangsu.china.com.cn/uploadfile/2015/1102/1446443026382534.jpg',
```

```
'http://n.sinaimg.cn/fashion/transform/20160704/flgG-fxtspsa6612705.jpg'))));
//将单个或者多个 Face 的 file 加入到一个 Person 中
var_dump ($client->faceAddFace('person2111', array('files'=>array('F:\pic\yang.jpg','F:\pic\yang2.jpg'))));
//将单个或者多个 Face 的文件内容加入到一个 Person 中
var_dump ($client->faceAddFace('person3111', array('buffers'=>array(file_get_contents('F:\pic\yang.jpg'),file_get_contents('F:\pic\yang2.jpg')))));
// 删除人脸，删除一个 Person 下的 face
var_dump ($client->faceDelFace('person1', array('12346',)));
//设置信息
var_dump ($client->faceSetInfo('person1', 'fanbing'));
//获取信息
var_dump ($client->faceGetInfo('person1'));
//获取组列表
var_dump ($client->faceGetGroupIds());
//获取人列表
var_dump ($client->faceGetPersonIds('group1'));
//获取人脸列表
var_dump ($client->faceGetFaceIds('person1'));
//获取人脸信息
var_dump ($client->faceGetFaceInfo('1704147773393235686'));
//删除个人
var_dump ($client->faceDelPerson('person11'));
```

人脸验证

给定一个 Face 和一个 Person，返回是否是同一个人的判断以及置信度。

```
//单个图片 URL
var_dump ($client->faceVerify('person1', array('url'=>'http://img3.a0bi.com/upload/ttq/20160814/1471155260063.png')));
//单个图片 file
var_dump ($client->faceVerify('person3111', array('file'=>'F:\pic\yang3.jpg')));
//单个图片内容
var_dump ($client->faceVerify('person3111', array('buffer'=>file_get_contents('F:\pic\yang3.jpg'))));
```

人脸对比

```
//两个对比图片的文件 URL
var_dump ($client->faceCompare(array('url'=>"http://imgsrc.baidu.com/baike/pic/item/5fdf8db1cb134954a4d833a0534e9258d0094a34.jpg"),
array('url'=>'http://a-ssl.duitang.com/uploads/item/201610/29/20161029215753_5cMTX.jpeg')));
//两个对比图片的文件 file
var_dump ($client->faceCompare(array('file'=>'F:\pic\yang.jpg'), array('file'=>'F:\pic\yang2.jpg')));
```

```
//两个对比图片的文件内容
var_dump ($client->faceCompare(array('file'=>'F:\pic\yang.jpg'), array('file'=>'F:\pic\yang2.jpg')));
```

人脸检索

对一张待识别的人脸图片，在一个或多个 group 中识别出最相似的 Top5 person 作为其身份返回，返回的 Top5 中按照相似度从大到小排列。

```
//单个文件 URL
var_dump ($client->faceIdentify('group1', array('url'=>'http://www.5djiaren.com/uploads/2016-07/22-141354_227.jpg')));
//单个文件 file
var_dump ($client->faceIdentify('group11', array('file'=>'F:\pic\yang3.jpg')));
//单个文件内容
var_dump ($client->faceIdentify('group11', array('buffer'=>file_get_contents('F:\pic\yang3.jpg'))));
```

人脸核身

身份证识别对比

```
//身份证 URL
var_dump ($client->faceIdCardCompare('xxxxxxxxxx', 'xxxxxxxx', array('url'=>'http://docs.ebdoor.com/Image/CompanyCertificate/1/16844.jpg')));
//身份证文件 file
var_dump ($client->faceIdCardCompare('xxxxxxxxxx', 'xxxxxxxxxx', array('file'=>'F:\pic\idcard.jpg')));
//身份证文件内容
var_dump ($client->faceIdCardCompare('xxxxxxxxxx', 'xxxxxxxxxx', array('buffer'=>file_get_contents('F:\pic\idcard.jpg'))));
```

活体检测 - 获取唇语验证码

```
$obj = $client->faceLiveGetFour();
var_dump ($obj);
$validate_data = $obj['data']['validate_data'];
```

活体检测 - 视频与用户照片的比对

```
var_dump ($client->faceLiveDetectFour($validate_data, array('file'=>'F:\pic\ZOE_0171.mp4'), False, array('F:\pic\idcard.jpg')));
```

活体检测 - 视频身份信息核验

```
var_dump ($client->faceIdCardLiveDetectFour($validate_data, array('file'=>'F:\pic\ZOE_0171.mp4'), 'xxxxxxxxxx', 'xxxxxxxxxx'));
```

Node-SDK-图像识别

最近更新时间：2018-12-24 16:39:05

> ⚠ **注意：**
>
> 本 SDK 仅适用于2018年12月前发布的活动。
> 因引擎算法升级，请2018年12月1日后创建新活动的用户，使用 新版 SDK。

开发准备

SDK 获取

智能图像的 Node SDK 下载地址：Node-SDK 。

快速入门

在腾讯云申请业务的授权

授权包括： APPID 、SecretId 、 SecretKey ，目前只支持主账号及密钥进行调用。

安装使用

```
npm i --save image-node-sdk
```

以 OCR-身份证识别 为例，一般支持外链 url 或者本地读取图片文件这两种方式。

- 外链 url

```
const {
ImageClient
} = require('image-node-sdk');

let AppId = ''; // 腾讯云 AppId
let SecretId = ''; // 腾讯云 SecretId
let SecretKey = ''; // 腾讯云 SecretKey

let idCardImageUrl = 'http://images.cnitblog.com/blog/454646/201306/07090518-029ff26fac014d72a7786937e8319c78.jpg';
let imgClient = new ImageClient({ AppId, SecretId, SecretKey });
imgClient.ocrIdCard({
```

```
data: {
url_list: [idCardImageUrl]
}
}).then((result) => {
console.log(result.body)
}).catch((e) => {
console.log(e);
});
```

- 读取本地文件

```
const fs = require('fs');
const path = require('path');
const {
ImageClient
} = require('image-node-sdk');

let AppId = ''; // 腾讯云 AppId
let SecretId = ''; // 腾讯云 SecretId
let SecretKey = ''; // 腾讯云 SecretKey

let imgClient = new ImageClient({ AppId, SecretId, SecretKey });
imgClient.ocrIdCard({
formData: {
card_type: 0,
image: fs.createReadStream(path.join(__dirname, './idcard.jpg'))
},
headers: {
'content-type': 'multipart/form-data'
}
}).then((result) => {
console.log(result.body)
}).catch((e) => {
console.log(e);
});
```

如果想运行，example/index.js 下面的例子，请先在项目根目录新建 config/index.js 文件，并按以下格式写下配置：

```
const ProxyUrl = ''; // 可填公司代理
const AppId = ''; // 腾讯云 AppId
const SecretId = ''; // 腾讯云 SecretId
const SecretKey = ''; // 腾讯云 SecretKey
```

```
exports.ProxyUrl = ProxyUrl;
exports.AppId = AppId;
exports.SecretId = SecretId;
exports.SecretKey = SecretKey;
```

然后运行：

```
npm run example
```

支持功能

- 信息认证
  - 身份证信息认证 - authIdCard
- 人脸识别
  - 多脸检索 - faceMultiple
  - 人脸检测与分析 - faceDetect
  - 五官定位 - faceShape
  - 个体信息管理-个体创建 - faceNewPerson
  - 个体信息管理-删除个体 - faceDelPerson
  - 个体信息管理-增加人脸 - faceAddFace
  - 个体信息管理-删除人脸 - faceDelFace
  - 个体信息管理-设置信息 - faceSetInfo
  - 个体信息管理-获取信息 - faceGetInfo
  - 个体信息管理-获取组列表 - faceGetGpIds
  - 个体信息管理-获取人列表 - faceGetPersonIds
  - 个体信息管理-获取人脸列表 - faceGetFaceIds
  - 个体信息管理-获取人脸信息 - faceGetFaceInfo
  - 个体信息管理-新增组信息 - faceAddGPIds
  - 个体信息管理-删除组信息 - faceDelGPIds
  - 人脸验证 - faceVerify
  - 人脸检索 - faceIdentify
  - 人脸对比 - faceCompare
- 文字识别 OCR
  - 手写体识别 - ocrHandWriting

- 身份证识别 - ocrIdCard
- 营业执照识别 - ocrBizLicense
- 行驶证驾驶证识别 - ocrDrivingLicence
- 车牌号识别 - ocrPlate
- 通用印刷体识别 - ocrGeneral
- 银行卡识别 - ocrBankCard
- 名片识别（V2) - ocrBizCard

- 图片识别
  - 图片标签 - imgTagDetect
  - 图片鉴黄 - imgPornDetect

- 人脸核身
  - 人脸静态活体检测 - faceLiveDetectPic
  - 唇语活体检测视频身份信息核验 - faceIdCardLiveDetectFour
  - 活体检测—获取唇语验证码 - faceLiveGetFour
  - 活体检测视频与用户照片的对比 - faceLiveDetectFour
  - 用户上传照片身份信息核验 - faceIdCardCompare

- 人脸融合
  - 人脸融合 - faceFusion