ALIBABA CLOUD

阿里云

内容安全

SDK参考

文档版本：20220706

阿里云

法律声明

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

1. 您应当通过阿里云网站或阿里云提供的其他授权通道下载、获取本文档，且仅能用于自身的合法合规的业务活动。本文档的内容视为阿里云的保密信息，您应当严格遵守保密义务；未经阿里云事先书面同意，您不得向任何第三方披露本手册内容或提供给任何第三方使用。

2. 未经阿里云事先书面许可，任何单位、公司或个人不得擅自摘抄、翻译、复制本文档内容的部分或全部，不得以任何方式或途径进行传播和宣传。

3. 由于产品版本升级、调整或其他原因，本文档内容有可能变更。阿里云保留在没有任何通知或者提示下对本文档的内容进行修改的权利，并在阿里云授权通道中不时发布更新后的用户文档。您应当实时关注用户文档的版本变更并通过阿里云授权渠道下载、获取最新版的用户文档。

4. 本文档仅作为用户使用阿里云产品及服务的参考性指引，阿里云以产品及服务的“现状”、“有缺陷”和“当前功能”的状态提供本文档。阿里云在现有技术的基础上尽最大努力提供相应的介绍及操作指引，但阿里云在此明确声明对本文档内容的准确性、完整性、适用性、可靠性等不作任何明示或暗示的保证。任何单位、公司或个人因为下载、使用或信赖本文档而发生任何差错或经济损失的，阿里云不承担任何法律责任。在任何情况下，阿里云均不对任何间接性、后果性、惩戒性、偶然性、特殊性或刑罚性的损害，包括用户使用或信赖本文档而遭受的利润损失，承担责任（即使阿里云已被告知该等损失的可能性）。

5. 阿里云网站上所有内容，包括但不限于著作、产品、图片、档案、资讯、资料、网站架构、网站画面的安排、网页设计，均由阿里云和/或其关联公司依法拥有其知识产权，包括但不限于商标权、专利权、著作权、商业秘密等。非经阿里云和/或其关联公司书面同意，任何人不得擅自使用、修改、复制、公开传播、改变、散布、发行或公开发表阿里云网站、产品程序或内容。此外，未经阿里云事先书面同意，任何人不得为了任何营销、广告、促销或其他目的使用、公布或复制阿里云的名称（包括但不限于单独为或以组合形式包含“阿里云”、“Aliyun”、“万网”等阿里云和/或其关联公司品牌，上述品牌的附属标志及图案或任何类似公司名称、商号、商标、产品或服务名称、域名、图案标示、标志、标识或通过特定描述使第三方能够识别阿里云和/或其关联公司）。

6. 如若发现本文档存在任何错误，请与阿里云取得直接联系。

通用约定

格式	说明	样例
危险	该类警示信息将导致系统重大变更甚至故障，或者导致人身伤害等结果。	危险 重置操作将丢失用户配置数据。
警告	该类警示信息可能会导致系统重大变更甚至故障，或者导致人身伤害等结果。	警告 重启操作将导致业务中断，恢复业务时间约十分钟。
注意	用于警示信息、补充说明等，是用户必须了解的内容。	注意 权重设置为0，该服务器不会再接受新请求。
说明	用于补充说明、最佳实践、窍门等，不是用户必须了解的内容。	说明 您也可以通过按Ctrl+A选中全部文件。
>	多级菜单递进。	单击**设置**> **网络**> **设置网络类型**。
粗体	表示按键、菜单、页面名称等UI元素。	在**结果确认**页面，单击**确定**。
`Courier字体`	命令或代码。	执行 `cd /d C:/window` 命令，进入Windows系统文件夹。
斜体	表示参数、变量。	`bae log list --instanceid` *Instance_ID*
[] 或者 [a|b]	表示可选项，至多选择一个。	`ipconfig [-all|-t]`
{} 或者 {a|b}	表示必选项，至多选择一个。	`switch {active|stand}`

目录

1.SDK概览

本文介绍了如何使用内容安全服务提供的内容检测API SDK。

SDK使用说明

在使用SDK前，您需要阅读内容检测API文档，了解各个接口的具体功能。

- 我们将图像检测相关的功能封装成一个接口（例如图片鉴黄、图片涉政暴恐检测、图片OCR、图片Logo检测等），并提供以下两种调用方式：
  - 图片同步检测接口：支持对多张图片进行检测，同步返回所有检测结果，建议一次检测一张图片。
  - 图片异步检测接口：支持对批量图片进行检测，接口将针对每一张图片返回一个taskId，您需要在提交检测任务后，通过taskId获取检测结果，对于批量图片检测，推荐使用该方式。
- 我们将视频检测相关的功能封装成一个接口（例如视频鉴黄、视频涉政暴恐检测、 视频Logo检测等），并提供以下两种调用方式：
  - 视频同步检测接口：只支持用户自己将视频截成图片帧序列，传递图片序列进行检测，不推荐使用该方式。
  - 视频异步检测接口：支持用户传递视频进行检测，您需要在提交检测任务后，通过taskId获取检测结果或者通过设置回调接口接收检测的结果回调通知，推荐您使用该方式进行视频内容检测。
- 语音反垃圾：语音垃圾内容检测SDK支持语音流和语音文件的检测，目前只有异步检测接口，您需要在提交检测任务后，通过taskId获取检测结果或者通过设置回调接口接收检测的结果回调通知。
- 文本反垃圾：文本反垃圾只有同步检测接口，您可以在一次请求中检测一条或者多条文本。

> **说明** 在一个接口（例如图片检测）中调用多个场景进行检测时，按照“每个场景的计费单价×检测的内容量”进行计费。

开发准备

- 准备各语言SDK依赖的开发环境。

  访问阿里云开发工具包（SDK），选择您的开发语言，下载并准备阿里云内容安全SDK的依赖环境。

- 内容安全SDK支持以下语言或环境：
  - Java SDK

- PHP SDK
- Python SDK

- 下载SDK使用代码示例。

  单击 green-sdk-sample_doc下载SDK代码示例。

  上述代码示例里面包含完整的Java、PHP、Python调用示例，供您参考。

- 参考第三方SDK。

  如果您使用除Java、PHP、Python以外的开发语言，推荐您通过HTTP请求直接调用内容检测API；我们也收集了一些第三方开发者编写的内容安全SDK，供您参考。内容安全第三方SDK包括以下语言：c#、c++、nodejs、python（3.5）、go。具体内容，请参见其他语言 SDK。

  > **说明** 对第三方SDK，阿里云不提供后续维护，只作列举参考。

技术答疑

如果您在接入API或SDK的过程中遇到问题，可以搜索钉钉群号（35573806）加入内容安全技术答疑群。加群时请备注您的公司名称+职位类型（研发、产品或者审核），例如XXX公司+研发。

2.Java SDK
2.1. 安装

本文基于aliyun-java-sdk-green 3.6.5介绍如何安装Java SDK。

环境准备

环境要求：使用Java 1.6及以上版本。

> **说明** 您可以执行命令 `java -version` 查看Java版本。

安装SDK

您只需在*pom.xml*中加入相应依赖，就可以在Maven工程中使用*aliyun-java-sdk-green*。以3.6.5版本为例，在 `<dependencies>` 中引入以下内容：

```
<dependency>
    <groupId>com.aliyun</groupId>
    <artifactId>aliyun-java-sdk-core</artifactId>
    <version>4.1.1</version>
</dependency>
<dependency>
    <groupId>com.aliyun</groupId>
    <artifactId>aliyun-java-sdk-green</artifactId>
    <version>3.6.5</version>
</dependency>
<dependency>
    <groupId>com.alibaba</groupId>
    <artifactId>fastjson</artifactId>
    <version>1.2.51</version>
</dependency>
<dependency>
    <groupId>com.aliyun.oss</groupId>
    <artifactId>aliyun-sdk-oss</artifactId>
    <version>2.8.3</version>
</dependency>
```

Java示例中编写的代码有部分内容基于Java依赖包，用于读取文件以及字符串转化。您可按需在 `<dependencies>` 中引入以下内容：

```
<dependency>
    <groupId>commons-io</groupId>
    <artifactId>commons-io</artifactId>
    <version>2.4</version>
</dependency>
<dependency>
    <groupId>commons-codec</groupId>
    <artifactId>commons-codec</artifactId>
    <version>1.10</version>
</dependency>
```

如果您需要检测本地文件或者二进制文件，请下载并在项目工程中引入Extension.Uploader工具类。

2.2. 初始化

IAcsClient是aliyun-java-sdk-green的Java客户端。使用aliyun-java-sdk-green Java SDK发起请求前，您需要初始化一个IAcsClient实例，并根据需要修改IClientProfile的配置项。

背景信息

新建IAcsClient时，需要指定服务地域（Region）和阿里云账号的AccessKey信息（包括AccessKey ID和AccessKey Secret）。

- 关于服务地域（Region）的更多信息，请参见接入区域。
- 关于阿里云账号的AccessKey的更多信息，请参见获取AccessKey。

> **说明** 新加坡地域仅支持部分算法模型。

图片、语音、视频、文本检测的IAcsClient

支持的地域（Region）包括：

- cn-shanghai：华东2（上海）
- cn-beijing：华北2（北京）
- ap-southeast-1：新加坡

> **说明** 新加坡地域仅支持部分算法模型。

您可以使用以下代码创建IAcsClient，用于图片、语音、视频、文本检测：

```
/**
      * ALIYUN_ACCESS_KEY_ID和ALIYUN_ACCESS_KEY_SECRET中传入您自己的AccessKey信息。
      * REGION_ID可选值：cn-shanghai、cn-beijing、ap-southeast-1。其他地域暂不支持，请勿使用。
      * 阿里云主账号AccessKey拥有所有API的访问权限，风险很高。强烈建议您创建并使用RAM子用户进行API访问或日常运维。
      */
     String ALIYUN_ACCESS_KEY_ID = "<yourAccessKeyId>";
     String ALIYUN_ACCESS_KEY_SECRET = "<yourAccessKeySecret>";
     String REGION_ID = "cn-shanghai";
     IClientProfile profile = DefaultProfile.getProfile(REGION_ID, ALIYUN_ACCESS_KEY_ID, ALIYUN_ACCESS_KEY_SECRET);
     IAcsClient recognitionClient = new DefaultAcsClient(profile)
```

自定义图片库、关键词词库、相似文本库的IAcsClient

仅支持华东2（上海）地域，Region ID为cn-shanghai。您只需通过cn-shanghai地域添加IAcsClient，其他地域的服务端会自动同步数据。

您可以使用以下代码创建IAcsClient，用于管理自定义图片库、自定义关键词词库、自定义相似文本库：

```
/**
      * ALIYUN_ACCESS_KEY_ID和ALIYUN_ACCESS_KEY_SECRET中传入您自己的AccessKey信息。
      * REGION_ID可选值：cn-shanghai。其他地域暂不支持，请勿使用。
      * 阿里云主账号AccessKey拥有所有API的访问权限，风险很高。强烈建议您创建并使用RAM子用户进行API访问或日常运维。
      */
     String ALIYUN_ACCESS_KEY_ID = "<yourAccessKeyId>";
     String ALIYUN_ACCESS_KEY_SECRET = "<yourAccessKeySecret>";
     String REGION_ID = "cn-shanghai";
     IClientProfile profile = DefaultProfile.getProfile(REGION_ID, ALIYUN_ACCESS_KEY_ID, ALIYUN_ACCESS_KEY_SECRET);
     IAcsClient managementClient = new DefaultAcsClient(profile)
```

2.3. 内容审核（机审）

2.3.1. 图片审核

本文介绍了如何使用Java SDK图片审核接口，检测图片中是否包含风险内容。

功能描述

图片审核支持同步检测和异步检测两种方式。

- 同步检测实时返回检测结果。关于参数的详细信息，请参见同步检测。
- 异步检测需要您轮询结果或者通过callback回调通知获取检测结果。关于参数的详细信息，请参见异步检测。

前提条件

- 安装Java依赖。关于安装Java依赖的具体操作，请参见安装Java依赖。

  > **说明** 请一定按照安装Java依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

（推荐）提交图片同步检测任务

接口	描述	支持的地域
ImageSyncScanRequest	提交图片同步检测任务，对图片进行多个风险场景的识别，包括色情、暴恐涉政、广告、二维码、不良场景、Logo（商标台标）识别。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

- 传图片URL进行检测

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
```

```
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.green.model.v20180509.ImageSyncScanRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.http.MethodType;
import com.aliyuncs.http.ProtocolType;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import java.util.*;
public class Main {
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile
            .getProfile("cn-shanghai", "请填写您的AccessKey ID", "请填写您的AccessKey Secret
");
        DefaultProfile
            .addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        ImageSyncScanRequest imageSyncScanRequest = new ImageSyncScanRequest();
        // 指定API返回格式。
        imageSyncScanRequest.setAcceptFormat(FormatType.JSON);
        // 指定请求方法。
        imageSyncScanRequest.setMethod(MethodType.POST);
        imageSyncScanRequest.setEncoding("utf-8");
        // 支持HTTP和HTTPS。
        imageSyncScanRequest.setProtocol(ProtocolType.HTTP);
        JSONObject httpBody = new JSONObject();
        /**
         * 设置要检测的风险场景。计费依据此处传递的场景计算。
         * 一次请求中可以同时检测多张图片，每张图片可以同时检测多个风险场景，计费按照场景计算。
         * 例如，检测2张图片，场景传递porn和terrorism，计费会按照2张图片鉴黄，2张图片暴恐检测计算
。
         * porn：表示鉴黄场景。
         */
        httpBody.put("scenes", Arrays.asList("porn"));
        /**
         * 设置待检测图片。一张图片对应一个task。
         * 多张图片同时检测时，处理的时间由最后一个处理完的图片决定。
         * 通常情况下批量检测的平均响应时间比单张检测的要长。一次批量提交的图片数越多，响应时间被拉
长的概率越高。
         * 这里以单张图片检测作为示例，如果是批量图片检测，请自行构建多个task。
         */
        JSONObject task = new JSONObject();
        task.put("dataId", UUID.randomUUID().toString());
        // 设置图片链接。
        task.put("url", "http://www.aliyundoc.com/xxx.test.jpg");
        task.put("time", new Date());
        httpBody.put("tasks", Arrays.asList(task));
        imageSyncScanRequest.setHttpContent(org.apache.commons.codec.binary.StringUtils.g
etBytesUtf8(httpBody.toJSONString()),
            "UTF-8", FormatType.JSON);
        /**
         * 请设置超时时间。服务端全链路处理超时时间为10秒，请做相应设置。
         * 如果您设置的ReadTimeout小于服务端处理的时间，程序中会获得一个ReadTimeout异常。
```

```
         */
        imageSyncScanRequest.setConnectTimeout(3000);
        imageSyncScanRequest.setReadTimeout(10000);
        HttpResponse httpResponse = null;
        try {
            httpResponse = client.doAction(imageSyncScanRequest);
        } catch (Exception e) {
            e.printStackTrace();
        }
        // 服务端接收到请求，完成处理后返回的结果。
        if (httpResponse != null && httpResponse.isSuccess()) {
            JSONObject scrResponse = JSON.parseObject(org.apache.commons.codec.binary.Str
ingUtils.newStringUtf8(httpResponse.getHttpContent()));
            System.out.println(JSON.toJSONString(scrResponse, true));
            int requestCode = scrResponse.getIntValue("code");
            // 每一张图片的检测结果。
            JSONArray taskResults = scrResponse.getJSONArray("data");
            if (200 == requestCode) {
                for (Object taskResult : taskResults) {
                    // 单张图片的处理结果。
                    int taskCode = ((JSONObject) taskResult).getIntValue("code");
                    // 图片对应检测场景的处理结果。如果是多个场景，则会有每个场景的结果。
                    JSONArray sceneResults = ((JSONObject) taskResult).getJSONArray("resu
lts");
                    if (200 == taskCode) {
                        for (Object sceneResult : sceneResults) {
                            String scene = ((JSONObject) sceneResult).getString("scene");
                            String suggestion = ((JSONObject) sceneResult).getString("sug
gestion");
                            // 根据scene和suggestion做相关处理。
                            // 根据不同的suggestion结果做业务上的不同处理。例如，将违规数据删除
等。
                            System.out.println("scene = [" + scene + "]");
                            System.out.println("suggestion = [" + suggestion + "]");
                        }
                    } else {
                        // 单张图片处理失败，原因视具体的情况详细分析。
                        System.out.println("task process fail. task response:" + JSON.toJ
SONString(taskResult));
                    }
                }
            } else {
                /**
                 * 表明请求整体处理失败，原因视具体的情况详细分析。
                 */
                System.out.println("the whole image scan request failed. response:" + JSO
N.toJSONString(scrResponse));
            }
        }
    }
}
```

- 传本地图片文件进行检测

```
import com.alibaba.fastjson.JSON;
```

```
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.green.extension.uploader.ClientUploader;
import com.aliyuncs.green.model.v20180509.ImageSyncScanRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.http.MethodType;
import com.aliyuncs.http.ProtocolType;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import java.util.*;
public class Main {
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile
            .getProfile("cn-shanghai", "请填写您的AccessKey ID", "请填写您的AccessKey Secret
");
        DefaultProfile
            .addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        ImageSyncScanRequest imageSyncScanRequest = new ImageSyncScanRequest();
        // 指定API返回格式。
        imageSyncScanRequest.setAcceptFormat(FormatType.JSON);
        // 指定请求方法。
        imageSyncScanRequest.setMethod(MethodType.POST);
        imageSyncScanRequest.setEncoding("utf-8");
        // 支持HTTP和HTTPS。
        imageSyncScanRequest.setProtocol(ProtocolType.HTTP);
        JSONObject httpBody = new JSONObject();
        /**
         * 设置要检测的场景。计费依据此处传递的场景计算。
         * 一次请求中可以同时检测多张图片，每张图片可以同时检测多个风险场景，计费按照场景计算。
         * 例如：检测2张图片，场景传递porn和terrorism，计费会按照2张图片鉴黄，2张图片暴恐检测计算
。
         * porn：表示色情场景检测。
         */
        httpBody.put("scenes", Arrays.asList("porn"));
        /**
         * 如果您要检测的文件存于本地服务器上，可以通过下述代码片生成URL。
         * 再将返回的URL作为图片地址传递到服务端进行检测。
         */
        String url = null;
        ClientUploader clientUploader = ClientUploader.getImageClientUploader(profile, fa
lse);
        try{
            url = clientUploader.uploadFile("d:/test.jpg");
        }catch (Exception e){
            e.printStackTrace();
        }
        /**
         * 设置待检测图片。一张图片对应一个task。
         * 多张图片同时检测时，处理的时间由最后一个处理完的图片决定。
         * 通常情况下批量检测的平均响应时间比单张检测的要长，一次批量提交的图片数越多，响应时间被拉
```

```
长的概率越高。
         * 这里以单张图片检测作为示例。如果是批量图片检测，请自行构建多个task。
         */
        JSONObject task = new JSONObject();
        task.put("dataId", UUID.randomUUID().toString());
        // 设置图片链接为上传后的URL。
        task.put("url", url);
        task.put("time", new Date());
        httpBody.put("tasks", Arrays.asList(task));
        imageSyncScanRequest.setHttpContent(org.apache.commons.codec.binary.StringUtils.g
etBytesUtf8(httpBody.toJSONString()),
            "UTF-8", FormatType.JSON);
        /**
         * 请设置超时时间。服务端全链路处理超时时间为10秒，请做相应设置。
         * 如果您设置的ReadTimeout小于服务端处理的时间，程序中会获得一个ReadTimeout异常。
         */
        imageSyncScanRequest.setConnectTimeout(3000);
        imageSyncScanRequest.setReadTimeout(10000);
        HttpResponse httpResponse = null;
        try {
            httpResponse = client.doAction(imageSyncScanRequest);
        } catch (Exception e) {
            e.printStackTrace();
        }
        // 服务端接收到请求，并完成处理后返回的结果。
        if (httpResponse != null && httpResponse.isSuccess()) {
            JSONObject scrResponse = JSON.parseObject(org.apache.commons.codec.binary.Str
ingUtils.newStringUtf8(httpResponse.getHttpContent()));
            System.out.println(JSON.toJSONString(scrResponse, true));
            int requestCode = scrResponse.getIntValue("code");
            // 每一张图片的检测结果。
            JSONArray taskResults = scrResponse.getJSONArray("data");
            if (200 == requestCode) {
                for (Object taskResult : taskResults) {
                    // 单张图片的处理结果。
                    int taskCode = ((JSONObject) taskResult).getIntValue("code");
                    // 图片对应检测场景的处理结果。如果是多个场景，则会有每个场景的结果。
                    JSONArray sceneResults = ((JSONObject) taskResult).getJSONArray("resu
lts");
                    if (200 == taskCode) {
                        for (Object sceneResult : sceneResults) {
                            String scene = ((JSONObject) sceneResult).getString("scene");
                            String suggestion = ((JSONObject) sceneResult).getString("sug
gestion");
                            // 根据scene和suggestion做相关处理。
                            // 根据不同的suggestion结果做业务上的不同处理。例如，将违规数据删除
等。
                            System.out.println("scene = [" + scene + "]");
                            System.out.println("suggestion = [" + suggestion + "]");
                        }
                    } else {
                        // 单张图片处理失败，原因视具体的情况详细分析。
                        System.out.println("task process fail. task response:" + JSON.toJ
SONString(taskResult));
```

```
                    }
                }
            } else {
                /**
                 * 表明请求整体处理失败，原因视具体的情况详细分析。
                 */
                System.out.println("the whole image scan request failed. response:" + JSO
N.toJSONString(scrResponse));
            }
        }
    }
}
```

- 传图片二进制内容进行检测

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.green.extension.uploader.ClientUploader;
import com.aliyuncs.green.model.v20180509.ImageSyncScanRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.http.MethodType;
import com.aliyuncs.http.ProtocolType;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import org.apache.commons.io.FileUtils;
import java.io.File;
import java.util.*;
public class Main {
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile
            .getProfile("cn-shanghai", "请填写您的AccessKey ID", "请填写您的AccessKey Secret
");
        DefaultProfile
            .addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        ImageSyncScanRequest imageSyncScanRequest = new ImageSyncScanRequest();
        // 指定API返回格式。
        imageSyncScanRequest.setAcceptFormat(FormatType.JSON);
        // 指定请求方法。
        imageSyncScanRequest.setMethod(MethodType.POST);
        imageSyncScanRequest.setEncoding("utf-8");
        // 支持HTTP和HTTPS。
        imageSyncScanRequest.setProtocol(ProtocolType.HTTP);
        JSONObject httpBody = new JSONObject();
        /**
         * 设置要检测的场景。计费依据此处传递的场景计算。
         * 一次请求中可以同时检测多张图片，每张图片可以同时检测多个风险场景，计费按照场景计算。
         * 例如，检测2张图片，场景传递porn和terrorism，计费会按照2张图片鉴黄，2张图片暴恐检测计算
。
         * porn：表示色情场景检测。
         */
```

```
        httpBody.put("scenes", Arrays.asList("porn"));
        /**
         * 如果您要检测的文件存于本地服务器上，可以通过下述代码片生成URL。
         * 再将返回的URL作为图片地址传递到服务端进行检测。
         */
        ClientUploader clientUploader = ClientUploader.getImageClientUploader(profile, fa
lse);
        byte[] imageBytes = null;
        String url = null;
        try{
            // 这里读取本地文件作为二进制数据，当做输入做为示例。实际使用中请直接替换成您的图片二进
制数据。
            imageBytes = FileUtils.readFileToByteArray(new File("/Users/01fb4ab6420b5f346
23e13b82b51ef87.jpg"));
            // 上传到服务端。
            url = clientUploader.uploadBytes(imageBytes);
        }catch (Exception e){
            e.printStackTrace();
        }
        /**
         * 设置待检测图片。一张图片对应一个task。
         * 多张图片同时检测时，处理的时间由最后一个处理完的图片决定。
         * 通常情况下批量检测的平均响应时间比单张检测的要长，一次批量提交的图片数越多，响应时间被拉
长的概率越高。
         * 这里以单张图片检测作为示例。如果是批量图片检测，请自行构建多个task。
         */
        JSONObject task = new JSONObject();
        task.put("dataId", UUID.randomUUID().toString());
        // 设置图片链接为上传后的URL。
        task.put("url", url);
        task.put("time", new Date());
        httpBody.put("tasks", Arrays.asList(task));
        imageSyncScanRequest.setHttpContent(org.apache.commons.codec.binary.StringUtils.g
etBytesUtf8(httpBody.toJSONString()),
            "UTF-8", FormatType.JSON);
        /**
         * 请设置超时时间。服务端全链路处理超时时间为10秒，请做相应设置。
         * 如果您设置的ReadTimeout小于服务端处理的时间，程序中会获得一个ReadTimeout异常。
         */
        imageSyncScanRequest.setConnectTimeout(3000);
        imageSyncScanRequest.setReadTimeout(10000);
        HttpResponse httpResponse = null;
        try {
            httpResponse = client.doAction(imageSyncScanRequest);
        } catch (Exception e) {
            e.printStackTrace();
        }
        // 服务端接收到请求，完成处理后返回的结果。
        if (httpResponse != null && httpResponse.isSuccess()) {
            JSONObject scrResponse = JSON.parseObject(org.apache.commons.codec.binary.Str
ingUtils.newStringUtf8(httpResponse.getHttpContent()));
            System.out.println(JSON.toJSONString(scrResponse, true));
            int requestCode = scrResponse.getIntValue("code");
            // 每一张图片的检测结果。
```

```
            JSONArray taskResults = scrResponse.getJSONArray("data");
            if (200 == requestCode) {
                for (Object taskResult : taskResults) {
                    // 单张图片的处理结果。
                    int taskCode = ((JSONObject) taskResult).getIntValue("code");
                    // 图片对应检测场景的处理结果。如果是多个场景，则会有每个场景的结果。
                    JSONArray sceneResults = ((JSONObject) taskResult).getJSONArray("resu
lts");
                    if (200 == taskCode) {
                        for (Object sceneResult : sceneResults) {
                            String scene = ((JSONObject) sceneResult).getString("scene");
                            String suggestion = ((JSONObject) sceneResult).getString("sug
gestion");
                            // 根据scene和suggestion做相关处理。
                            // 根据不同的suggestion结果做业务上的不同处理。例如，将违规数据删除
等。
                            System.out.println("scene = [" + scene + "]");
                            System.out.println("suggestion = [" + suggestion + "]");
                        }
                    } else {
                        // 单张图片处理失败，原因视具体的情况详细分析。
                        System.out.println("task process fail. task response:" + JSON.toJ
SONString(taskResult));
                    }
                }
            } else {
                /**
                 * 表明请求整体处理失败，原因视具体的情况详细分析。
                 */
                System.out.println("the whole image scan request failed. response:" + JSO
N.toJSONString(scrResponse));
            }
        }
    }
}
```

提交图片异步检测任务

使用Java SDK对图片进行风险检测。通过异步请求提交检测任务，结果可以通过提交请求时设置callback，也可以通过ImageAsyncScanResultsRequest接口轮询。

异步检测支持的输入内容与同步检测一致（本地文件、图片URL、图片二进制流），以下仅以图片URL作为输入示例。

接口	描述	支持的地域
ImageAsyncScanRequest	提交图片异步检测任务，对图片进行多个风险场景的识别，包括色情、暴恐涉政、广告、二维码、不良场景、Logo（商标台标）识别。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.green.model.v20180509.ImageAsyncScanRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.http.MethodType;
import com.aliyuncs.http.ProtocolType;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import java.util.Arrays;
import java.util.Date;
import java.util.UUID;
public class Main {
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile.getProfile("cn-shanghai", "请填写您的AccessKey ID", "请填写您的AccessKey Secret");
        DefaultProfile.addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        ImageAsyncScanRequest imageAsyncScanRequest = new ImageAsyncScanRequest();
        // 指定API返回格式。
        imageAsyncScanRequest.setAcceptFormat(FormatType.JSON);
        // 指定请求方法。
        imageAsyncScanRequest.setMethod(MethodType.POST);
        imageAsyncScanRequest.setEncoding("utf-8");
        // 支持HTTP和HTTPS。
        imageAsyncScanRequest.setProtocol(ProtocolType.HTTP);
        JSONObject httpBody = new JSONObject();
        /**
         * 设置要检测的场景。计费依据此处传递的场景计算。
         * 一次请求中可以同时检测多张图片，每张图片可以同时检测多个风险场景，计费按照场景计算。
         * 例如：检测2张图片，场景传递porn和terrorism，计费会按照2张图片鉴黄，2张图片暴恐检测计算。
         * porn：表示色情场景检测。
         */
        httpBody.put("scenes", Arrays.asList("porn"));
        httpBody.put("callback", "http://www.aliyundoc.com/xxx.json");
        httpBody.put("seed", "yourPersonalSeed");
        /**
         * 设置待检测图片。一张图片对应一个task。最多支持50张图片同时检测，即需要构建50个task。
         * 多张图片同时检测时，处理的时间由最后一个处理完的图片决定。
         * 通常情况下批量检测的平均响应时间比单张检测的要长，一次批量提交的图片数越多，响应时间被拉长的概率越高。
         * 这里以单张图片检测作为示例。如果是批量图片检测，请自行构建多个task。
         */
        JSONObject task = new JSONObject();
        task.put("dataId", UUID.randomUUID().toString());
        // 设置图片链接。
        task.put("url", "http://www.aliyundoc.com/xxx.test.jpg");
        task.put("time", new Date());
        httpBody.put("tasks", Arrays.asList(task));
        imageAsyncScanRequest.setHttpContent(org.apache.commons.codec.binary.StringUtils.ge
```

```
tBytesUtf8(httpBody.toJSONString()),
            "UTF-8", FormatType.JSON);
        /**
         * 请设置超时时间。服务端全链路处理超时时间为10秒，请做相应设置。
         * 如果您设置的ReadTimeout小于服务端处理的时间，程序中会获得一个ReadTimeout异常。
         */
        imageAsyncScanRequest.setConnectTimeout(3000);
        imageAsyncScanRequest.setReadTimeout(10000);
        HttpResponse httpResponse = null;
        try {
            httpResponse = client.doAction(imageAsyncScanRequest);
        } catch (Exception e) {
            e.printStackTrace();
        }
        // 服务端接收到请求，完成处理后返回的结果。
        if (httpResponse != null && httpResponse.isSuccess()) {
            JSONObject scrResponse = JSON.parseObject(org.apache.commons.codec.binary.Strin
gUtils.newStringUtf8(httpResponse.getHttpContent()));
            System.out.println(JSON.toJSONString(scrResponse, true));
            int requestCode = scrResponse.getIntValue("code");
            // 每一张图片的检测结果。
            JSONArray taskResults = scrResponse.getJSONArray("data");
            if (200 == requestCode) {
                for (Object taskResult : taskResults) {
                    // 单张图片的处理结果。
                    int taskCode = ((JSONObject) taskResult).getIntValue("code");
                    // 图片要检测的场景的处理结果，如果是多个场景，则会有每个场景的结果。
                    JSONArray sceneResults = ((JSONObject) taskResult).getJSONArray("result
s");
                    if (200 == taskCode) {
                        // 保存taskId用于轮询结果。
                        System.out.println(((JSONObject)taskResult).getString("taskId"));
                    } else {
                        // 单张图片处理失败，原因视具体的情况详细分析。
                        System.out.println("task process fail. task response:" + JSON.toJSO
NString(taskResult));
                    }
                }
            } else {
                /**
                 * 表明请求整体处理失败，原因视具体的情况详细分析。
                 */
                System.out.println("the whole image scan request failed. response:" + JSON.
toJSONString(scrResponse));
            }
        }
    }
}
```

查询图片异步检测结果

接口	描述	支持的地域
ImageAsyncScanResultsRequest	查询图片异步检测任务的结果。支持同时查询多个检测任务的返回结果。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.green.model.v20180509.ImageAsyncScanRequest;
import com.aliyuncs.green.model.v20180509.ImageAsyncScanResultsRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.http.MethodType;
import com.aliyuncs.http.ProtocolType;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import java.util.*;
public class Main {
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile.getProfile("cn-shanghai", "请填写您的AccessKey ID", "请填写您的AccessKey Secret");
        DefaultProfile.addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        ImageAsyncScanResultsRequest imageAsyncScanResultsRequest = new ImageAsyncScanResultsRequest();
        // 指定API返回格式。
        imageAsyncScanResultsRequest.setAcceptFormat(FormatType.JSON);
        // 指定请求方法。
        imageAsyncScanResultsRequest.setMethod(MethodType.POST);
        imageAsyncScanResultsRequest.setEncoding("utf-8");
        // 支持HTTP和HTTPS。
        imageAsyncScanResultsRequest.setProtocol(ProtocolType.HTTP);
        List<String> taskIds = new ArrayList<String>();
        taskIds.add("img4hDosCHcrFk5jAMR80XWJN-1pZ@0p");
        imageAsyncScanResultsRequest.setHttpContent(JSON.toJSONString(taskIds).getBytes("UTF-8"), "UTF-8", FormatType.JSON);
        /**
         * 请务必设置超时时间。
         */
        imageAsyncScanResultsRequest.setConnectTimeout(3000);
        imageAsyncScanResultsRequest.setReadTimeout(6000);
        try {
            HttpResponse httpResponse = client.doAction(imageAsyncScanResultsRequest);
```

```
            if(httpResponse.isSuccess()){
                JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttpCo
ntent(), "UTF-8"));
                System.out.println(JSON.toJSONString(scrResponse, true));
                if (200 == scrResponse.getInteger("code")) {
                    JSONArray taskResults = scrResponse.getJSONArray("data");
                    for (Object taskResult : taskResults) {
                        if(200 == ((JSONObject)taskResult).getInteger("code")){
                            JSONArray sceneResults = ((JSONObject)taskResult).getJSONArray(
"results");
                            for (Object sceneResult : sceneResults) {
                                String scene = ((JSONObject)sceneResult).getString("scene")
;
                                String suggestion = ((JSONObject)sceneResult).getString("su
ggestion");
                                // 根据scene和suggestion做相关的处理。
                                // 根据不同的suggestion结果做业务上的不同处理。例如，将违规数据删
除等。
                            }
                        }else{
                            System.out.println("task process fail:" + ((JSONObject)taskResu
lt).getInteger("code"));
                        }
                    }
                } else {
                    System.out.println("detect not success. code:" + scrResponse.getInteger
("code"));
                }
            }else{
                System.out.println("response not success. status:" + httpResponse.getStatus
());
            }
        } catch (ServerException e) {
            e.printStackTrace();
        } catch (ClientException e) {
            e.printStackTrace();
        } catch (Exception e){
            e.printStackTrace();
        }
    }
}
```

图片检测结果反馈

如果您认为图片检测结果与您的预期不符，可以通过图片检测结果反馈接口，对检测结果进行纠正（系统会根据您反馈的结果，将图片添加到相似图片的黑名单库或者白名单库）。当您再次提交相似的内容进行检测时，以您反馈的label返回结果。

关于接口的说明，请参见检测结果反馈。

接口	描述	支持的Region

接口	描述	支持的Region
ImageScanFeedbackRequest	提交图片检测结果的反馈，以人工反馈的检测结果纠正算法检测结果。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.green.model.v20180509.ImageScanFeedbackRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.http.MethodType;
import com.aliyuncs.http.ProtocolType;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import java.util.Arrays;
public class Main {
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile
                .getProfile("cn-shanghai", "请填写您的AccessKey ID", "请填写您的AccessKey Secr
et");
        DefaultProfile
                .addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        ImageScanFeedbackRequest imageScanFeedbackRequest = new ImageScanFeedbackRequest();
        // 指定API返回格式。
        imageScanFeedbackRequest.setAcceptFormat(FormatType.JSON);
        // 指定请求方法。
        imageScanFeedbackRequest.setMethod(MethodType.POST);
        imageScanFeedbackRequest.setEncoding("utf-8");
        // 支持HTTP和HTTPS。
        imageScanFeedbackRequest.setProtocol(ProtocolType.HTTP);
        // scenes：检测场景，支持指定多个场景。
        // suggestion：期望的检测结果。pass：正常；block：违规。
        JSONObject httpBody = new JSONObject();
        httpBody.put("suggestion", "block");
        httpBody.put("scenes", Arrays.asList("ad", "terrorism"));
        httpBody.put("url", "图片链接");
        imageScanFeedbackRequest.setHttpContent(org.apache.commons.codec.binary.StringUtils
.getBytesUtf8(httpBody.toJSONString()),
                "UTF-8", FormatType.JSON);
        imageScanFeedbackRequest.setConnectTimeout(3000);
        imageScanFeedbackRequest.setReadTimeout(10000);
        try {
            HttpResponse httpResponse = client.doAction(imageScanFeedbackRequest);
```

```
            if (httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttpCo
ntent(), "UTF-8"));
                System.out.println(JSON.toJSONString(scrResponse, true));
                if (200 == scrResponse.getInteger("code")) {
                    // 调用成功。
                } else {
                    System.out.println("detect not success. code:" + scrResponse.getInteger
("code"));
                }
            } else {
                System.out.println("response not success. status:" + httpResponse.getStatus
());
            }
        } catch (ServerException e) {
            e.printStackTrace();
        } catch (ClientException e) {
            e.printStackTrace();
        }
    }
}
```

2.3.2. 视频审核

本文介绍了如何使用Java SDK视频审核接口，检测视频中是否包含风险内容。

功能描述

视频审核接口支持同步检测和异步检测两种方式。

- 同步检测只支持传递视频的截帧图片序列。关于参数的详细介绍，请参见同步检测。
- 异步检测（推荐）支持对原始视频或视频的截帧图片序列进行检测。关于参数的详细介绍，请参见异步检测。

前提条件

- 安装Java依赖。关于安装Java依赖的具体操作，请参见安装Java依赖。

  > **说明** 请一定按照安装Java依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

（推荐）提交视频异步检测任务

接口	描述	支持的地域
VideoAsyncScanRequest	提交视频异步检测任务，对视频进行多个风险场景的识别，包括色情、暴恐涉政、广告 、不良场景、Logo（商标台标）识别。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

- 传视频URL进行检测

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.green.model.v20180509.VideoAsyncScanRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.LinkedHashMap;
import java.util.List;
import java.util.Map;
import java.util.UUID;
public class Main {
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile.getProfile("cn-shanghai", "请填写您的AccessKey ID", "请填写您的AccessKey Secret");
        DefaultProfile.addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        VideoAsyncScanRequest videoAsyncScanRequest = new VideoAsyncScanRequest();
        videoAsyncScanRequest.setAcceptFormat(FormatType.JSON); // 指定API返回格式。
        videoAsyncScanRequest.setMethod(com.aliyuncs.http.MethodType.POST); // 指定请求方法。
        List<Map<String, Object>> tasks = new ArrayList<Map<String, Object>>();
        Map<String, Object> task = new LinkedHashMap<String, Object>();
        task.put("dataId", UUID.randomUUID().toString());
        task.put("url", "请填写公网可访问的视频HTTP、HTTPS URL地址");
        tasks.add(task);
        /**
         * 设置要检测的场景。计费是依据此处传递的场景计算。
         * 视频默认1秒截取一帧，您可以自行控制截帧频率。收费按照视频的截帧数量以及每一帧的检测场景计算。
         * 举例：1分钟的视频截帧60张，检测色情（对应场景参数porn）和暴恐涉政（对应场景参数terrorism）2个场景，收费按照60张色情+60张暴恐涉政进行计费。
         */
        JSONObject data = new JSONObject();
        data.put("scenes", Arrays.asList("porn", "terrorism"));
        data.put("tasks", tasks);
        data.put("callback", "http://www.aliyundoc.com/xxx.json");
        data.put("seed", "yourPersonalSeed");
        videoAsyncScanRequest.setHttpContent(data.toJSONString().getBytes("UTF-8"), "UTF-8", FormatType.JSON);
        /**
         * 请务必设置超时时间。
         */
        videoAsyncScanRequest.setConnectTimeout(3000);
```

```
        videoAsyncScanRequest.setReadTimeout(6000);
        try {
            HttpResponse httpResponse = client.doAction(videoAsyncScanRequest);
            if (httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttp
Content(), "UTF-8"));
                System.out.println(JSON.toJSONString(scrResponse, true));
                int requestCode = scrResponse.getIntValue("code");
                // 每一个视频的检测结果。
                JSONArray taskResults = scrResponse.getJSONArray("data");
                if (200 == requestCode) {
                    for (Object taskResult : taskResults) {
                        // 单个视频的处理结果。
                        int taskCode = ((JSONObject) taskResult).getIntValue("code");
                        if (200 == taskCode) {
                            // 保存taskId用于轮询结果。
                            System.out.println(((JSONObject) taskResult).getString("taskI
d"));
                        } else {
                            // 单个视频处理失败，原因视具体的情况详细分析。
                            System.out.println("task process fail. task response:" + JSON
.toJSONString(taskResult));
                        }
                    }
                } else {
                    /**
                     * 表明请求整体处理失败，原因视具体的情况详细分析。
                     */
                    System.out.println("the whole scan request failed. response:" + JSON.
toJSONString(scrResponse));
                }
            } else {
                System.out.println("response not success. status:" + httpResponse.getStat
us());
            }
        } catch (ServerException e) {
            e.printStackTrace();
        } catch (ClientException e) {
            e.printStackTrace();
        }
    }
}
```

- 传本地视频文件进行检测

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.green.extension.uploader.ClientUploader;
import com.aliyuncs.green.model.v20180509.VideoAsyncScanRequest;
import com.aliyuncs.http.FormatType;
```

```
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.LinkedHashMap;
import java.util.List;
import java.util.Map;
import java.util.UUID;
public class Main {
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile.getProfile("cn-shanghai", "请填写您的AccessKey ID", "请填写您的AccessKey Secret");
        DefaultProfile.addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        VideoAsyncScanRequest videoAsyncScanRequest = new VideoAsyncScanRequest();
        videoAsyncScanRequest.setAcceptFormat(FormatType.JSON); // 指定API返回格式。
        videoAsyncScanRequest.setMethod(com.aliyuncs.http.MethodType.POST); // 指定请求方法。
        /**
         * 如果您要检测的文件存储在本地服务器上，可以通过下述代码生成URL作为视频地址传递到服务端进行检测。
         */
        String url = null;
        ClientUploader uploader = ClientUploader.getVideoClientUploader(profile, false);
        try {
            url = uploader.uploadFile("您的本地文件的绝对路径");
        } catch (Exception e) {
            e.printStackTrace();
        }
        List<Map<String, Object>> tasks = new ArrayList<Map<String, Object>>();
        Map<String, Object> task = new LinkedHashMap<String, Object>();
        task.put("dataId", UUID.randomUUID().toString());
        task.put("url", url);
        tasks.add(task);
        /**
         * 设置要检测的场景。计费是依据此处传递的场景计算。
         * 视频默认1秒截取一帧，您可以自行控制截帧频率。收费按照视频的截帧数量以及每一帧的检测场景进行计算。
         * 举例：1分钟的视频截帧60张，检测色情（对应场景参数porn）和暴恐涉政（对应场景参数terrorism）2个场景，收费按照60张色情+60张暴恐涉政进行计算。
         */
        JSONObject data = new JSONObject();
        data.put("scenes", Arrays.asList("porn", "terrorism"));
        data.put("tasks", tasks);
        data.put("callback", "http://www.aliyundoc.com/xxx.json");
        data.put("seed", "yourPersonalSeed");
        videoAsyncScanRequest.setHttpContent(data.toJSONString().getBytes("UTF-8"), "UTF-8", FormatType.JSON);
        /**
         * 请务必设置超时时间。
         */
        videoAsyncScanRequest.setConnectTimeout(3000);
```

```
        videoAsyncScanRequest.setReadTimeout(10000);
        try {
            HttpResponse httpResponse = client.doAction(videoAsyncScanRequest);
            if (httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttp
Content(), "UTF-8"));
                System.out.println(JSON.toJSONString(scrResponse, true));
                int requestCode = scrResponse.getIntValue("code");
                // 每一个视频的检测结果。
                JSONArray taskResults = scrResponse.getJSONArray("data");
                if (200 == requestCode) {
                    for (Object taskResult : taskResults) {
                        // 单个视频的处理结果。
                        int taskCode = ((JSONObject) taskResult).getIntValue("code");
                        if (200 == taskCode) {
                            // 保存taskId用于轮询结果。
                            System.out.println(((JSONObject) taskResult).getString("taskI
d"));
                        } else {
                            // 单个视频处理失败，原因视具体的情况详细分析。
                            System.out.println("task process fail. task response:" + JSON
.toJSONString(taskResult));
                        }
                    }
                } else {
                    /**
                     * 表明请求整体处理失败，原因视具体的情况详细分析。
                     */
                    System.out.println("the whole image scan request failed. response:" +
JSON.toJSONString(scrResponse));
                }
            } else {
                System.out.println("response not success. status:" + httpResponse.getStat
us());
            }
        } catch (ServerException e) {
            e.printStackTrace();
        } catch (ClientException e) {
            e.printStackTrace();
        }
    }
}
```

- 传视频文件二进制数据进行检测

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.green.extension.uploader.ClientUploader;
import com.aliyuncs.green.model.v20180509.VideoAsyncScanRequest;
import com.aliyuncs.http.FormatType;
```

```
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import org.apache.commons.io.FileUtils;
import java.io.File;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.LinkedHashMap;
import java.util.List;
import java.util.Map;
import java.util.UUID;
public class Main {
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile.getProfile("cn-shanghai", "请填写您的AccessKey ID", "请填写您的AccessKey Secret");
        DefaultProfile.addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        VideoAsyncScanRequest videoAsyncScanRequest = new VideoAsyncScanRequest();
        videoAsyncScanRequest.setAcceptFormat(FormatType.JSON); // 指定API返回格式。
        videoAsyncScanRequest.setMethod(com.aliyuncs.http.MethodType.POST); // 指定请求方法。
        /**
         * 如果您要检测的文件存储在本地服务器上，可以通过下述代码生成URL作为视频地址传递到服务端进行检测。
         */
        ClientUploader uploader = ClientUploader.getVideoClientUploader(profile, false);
        byte[] videoBytes = null;
        String url = null;
        try {
            // 读取本地文件作为二进制数据当做输入做为示例。实际使用中请直接替换成您的视频二进制数据。
            videoBytes = FileUtils.readFileToByteArray(new File("您的本地文件路径"));
            // 上传到服务端。
            url = uploader.uploadBytes(videoBytes);
        } catch (Exception e) {
            System.out.println("upload file to server fail." + e.toString());
        }
        List<Map<String, Object>> tasks = new ArrayList<Map<String, Object>>();
        Map<String, Object> task = new LinkedHashMap<String, Object>();
        task.put("dataId", UUID.randomUUID().toString());
        task.put("url", url);
        tasks.add(task);
        /**
         * 设置要检测的场景。计费依据此处传递的场景计算。
         * 视频默认1秒截取一帧，您可以自行控制截帧频率。收费按照视频的截帧数量以及每一帧的检测场景进行计算。
         * 举例：1分钟的视频截帧60张，检测色情（对应场景参数porn）和暴恐涉政（对应场景参数terrorism）2个场景，收费按照60张色情+60张暴恐涉政进行计算。
         */
        JSONObject data = new JSONObject();
        data.put("scenes", Arrays.asList("porn", "terrorism"));
        data.put("tasks", tasks);
        data.put("callback", "http://www.aliyundoc.com/xxx.json");
```

```
        data.put("seed", "yourPersonalSeed");
        videoAsyncScanRequest.setHttpContent(data.toJSONString().getBytes("UTF-8"), "UTF-
8", FormatType.JSON);
        /**
         * 请务必设置超时时间。
         */
        videoAsyncScanRequest.setConnectTimeout(3000);
        videoAsyncScanRequest.setReadTimeout(10000);
        try {
            HttpResponse httpResponse = client.doAction(videoAsyncScanRequest);
            if (httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttp
Content(), "UTF-8"));
                System.out.println(JSON.toJSONString(scrResponse, true));
                int requestCode = scrResponse.getIntValue("code");
                // 每一个视频的检测结果。
                JSONArray taskResults = scrResponse.getJSONArray("data");
                if (200 == requestCode) {
                    for (Object taskResult : taskResults) {
                        // 单个视频的处理结果。
                        int taskCode = ((JSONObject) taskResult).getIntValue("code");
                        if (200 == taskCode) {
                            // 保存taskId用于轮询结果。
                            System.out.println(((JSONObject) taskResult).getString("taskI
d"));
                        } else {
                            // 单个视频处理失败，原因视具体的情况详细分析。
                            System.out.println("task process fail. task response:" + JSON
.toJSONString(taskResult));
                        }
                    }
                } else {
                    /**
                     * 表明请求整体处理失败，原因视具体的情况详细分析。
                     */
                    System.out.println("the whole scan request failed. response:" + JSON.
toJSONString(scrResponse));
                }
            } else {
                System.out.println("response not success. status:" + httpResponse.getStat
us());
            }
        } catch (ServerException e) {
            e.printStackTrace();
        } catch (ClientException e) {
            e.printStackTrace();
        }
    }
}
```

- 传视频直播流进行检测

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
```

```
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.green.model.v20180509.VideoAsyncScanRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.LinkedHashMap;
import java.util.List;
import java.util.Map;
public class Main {
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile.getProfile("cn-shanghai", "请填写您的AccessKey ID", "请填写您的AccessKey Secret");
        DefaultProfile.addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        VideoAsyncScanRequest videoAsyncScanRequest = new VideoAsyncScanRequest();
        videoAsyncScanRequest.setAcceptFormat(FormatType.JSON); // 指定API返回格式。
        videoAsyncScanRequest.setMethod(com.aliyuncs.http.MethodType.POST); // 指定请求方法。
        List<Map<String, Object>> tasks = new ArrayList<Map<String, Object>>();
        Map<String, Object> task = new LinkedHashMap<String, Object>();
        task.put("dataId", "业务ID");
        // URL填写直播流地址。
        task.put("url", "待检测视频链接地址");
        tasks.add(task);
        /**
         * 设置要检测的场景。计费依据此处传递的场景计算。
         * 视频默认1秒截取一帧，您可以自行控制截帧频率。收费按照视频的截帧数量以及每一帧的检测场景进行计算。
         * 举例：1分钟的视频截帧60张，检测色情（对应场景参数porn）和暴恐涉政（对应场景参数terrorism）2个场景，收费按照60张色情+60张暴恐涉政进行计算。
         */
        JSONObject data = new JSONObject();
        data.put("scenes", Arrays.asList("porn", "terrorism"));
        data.put("live", true);
        data.put("tasks", tasks);
        data.put("callback", "您的回调地址");
        data.put("seed", "yourPersonalSeed");
        videoAsyncScanRequest.setHttpContent(data.toJSONString().getBytes("UTF-8"), "UTF-8", FormatType.JSON);
        /**
         * 请务必设置超时时间。
         */
        videoAsyncScanRequest.setConnectTimeout(3000);
        videoAsyncScanRequest.setReadTimeout(10000);
        try {
            HttpResponse httpResponse = client.doAction(videoAsyncScanRequest);
            if (httpResponse.isSuccess()) {
```

```
                JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttp
Content(), "UTF-8"));
                System.out.println(JSON.toJSONString(scrResponse, true));
                int requestCode = scrResponse.getIntValue("code");
                // 每一个视频的检测结果。
                JSONArray taskResults = scrResponse.getJSONArray("data");
                if (200 == requestCode) {
                    for (Object taskResult : taskResults) {
                        // 单个视频的处理结果。
                        int taskCode = ((JSONObject) taskResult).getIntValue("code");
                        if (200 == taskCode) {
                            // 保存taskId用于轮询结果。
                            System.out.println(((JSONObject) taskResult).getString("taskI
d"));
                        } else {
                            // 单个视频处理失败，原因视具体的情况详细分析。
                            System.out.println("task process fail. task response:" + JSON
.toJSONString(taskResult));
                        }
                    }
                } else {
                    /**
                     * 表明请求整体处理失败，原因视具体的情况详细分析。
                     */
                    System.out.println("the whole scan request failed. response:" + JSON.
toJSONString(scrResponse));
                }
            } else {
                System.out.println("response not success. status:" + httpResponse.getStat
us());
            }
        } catch (ServerException e) {
            e.printStackTrace();
        } catch (ClientException e) {
            e.printStackTrace();
        }
    }
}
```

- 传视频语音进行综合检测

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.green.model.v20180509.VideoAsyncScanRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import java.util.ArrayList;
import java.util.Arrays;
```

```
import java.util.Arrays;
import java.util.LinkedHashMap;
import java.util.List;
import java.util.Map;
public class Main {
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile.getProfile("cn-shanghai", "请填写您的AccessKey ID", "请填写您的AccessKey Secret");
        DefaultProfile.addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        VideoAsyncScanRequest videoAsyncScanRequest = new VideoAsyncScanRequest();
        videoAsyncScanRequest.setAcceptFormat(FormatType.JSON); // 指定API返回格式。
        videoAsyncScanRequest.setMethod(com.aliyuncs.http.MethodType.POST); // 指定请求方法。
        List<Map<String, Object>> tasks = new ArrayList<Map<String, Object>>();
        Map<String, Object> task = new LinkedHashMap<String, Object>();
        task.put("dataId", "业务数据ID");
        // URL填写直播流地址。
        task.put("url", "待检测的视频链接地址");
        tasks.add(task);
        /**
         * 设置要检测的场景。计费依据此处传递的场景计算。
         * 视频默认1秒截取一帧，您可以自行控制截帧频率。收费按照视频的截帧数量以及每一帧的检测场景进行计算。
         * 举例：1分钟的视频截帧60张，检测色情（对应场景参数porn）和暴恐涉政（对应场景参数terrorism）2个场景，收费按照60张色情+60张暴恐涉政进行计费。
         */
        JSONObject data = new JSONObject();
        data.put("scenes", Arrays.asList("porn", "terrorism"));
        data.put("live", true);
        data.put("tasks", tasks);
        data.put("callback", "您的回调地址");
        data.put("seed", "yourPersonalSeed");
        /**
         * 如果检测视频画面的同时需要检测语音是否有风险内容，传递以下参数。
         * 注意语音的计费是按照时长进行，即该视频的时长*语音反垃圾的单价。
         */
        data.put("audioScenes", Arrays.asList("antispam"));
        videoAsyncScanRequest.setHttpContent(data.toJSONString().getBytes("UTF-8"), "UTF-8", FormatType.JSON);
        /**
         * 请务必设置超时时间。
         */
        videoAsyncScanRequest.setConnectTimeout(3000);
        videoAsyncScanRequest.setReadTimeout(10000);
        try {
            HttpResponse httpResponse = client.doAction(videoAsyncScanRequest);
            if (httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttpContent(), "UTF-8"));
                System.out.println(JSON.toJSONString(scrResponse, true));
                int requestCode = scrResponse.getIntValue("code");
                // 每一个视频的检测结果。
                JSONArray taskResults = scrResponse.getJSONArray("data");
```

```
                if (200 == requestCode) {
                    for (Object taskResult : taskResults) {
                        // 单个视频的处理结果。
                        int taskCode = ((JSONObject) taskResult).getIntValue("code");
                        if (200 == taskCode) {
                            // 保存taskId，用于轮询结果。
                            System.out.println(((JSONObject) taskResult).getString("taskI
d"));
                        } else {
                            // 单个视频处理失败，原因视具体的情况详细分析。
                            System.out.println("task process fail. task response:" + JSON
.toJSONString(taskResult));
                        }
                    }
                } else {
                    /**
                     * 表明请求整体处理失败，原因视具体的情况详细分析。
                     */
                    System.out.println("the whole scan request failed. response:" + JSON.
toJSONString(scrResponse));
                }
            } else {
                System.out.println("response not success. status:" + httpResponse.getStat
us());
            }
        } catch (ServerException e) {
            e.printStackTrace();
        } catch (ClientException e) {
            e.printStackTrace();
        }
    }
}
```

查询视频异步检测结果

接口	描述	支持的地域
VideoAsyncScanResultsRequest	查询视频异步检测任务的结果。 说明 该方法需要轮询结果，建议使用callback的方式获取结果。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
```

```
import com.aliyuncs.green.model.v20180509.VideoAsyncScanResultsRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import java.util.ArrayList;
import java.util.List;
public class Main {
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile.getProfile("cn-shanghai", "请填写您的AccessKey ID", "请填写您的AccessKey Secret");
        DefaultProfile.addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        VideoAsyncScanResultsRequest videoAsyncScanResultsRequest = new VideoAsyncScanResultsRequest();
        videoAsyncScanResultsRequest.setAcceptFormat(FormatType.JSON);
        List<String> taskList = new ArrayList<String>();
        // 这里添加要查询的taskId。提交任务的时候需要自行保存taskId。
        taskList.add("视频检测任务ID");
        videoAsyncScanResultsRequest.setHttpContent(JSON.toJSONString(taskList).getBytes("UTF-8"), "UTF-8", FormatType.JSON);
        /**
         * 请务必设置超时时间。
         */
        videoAsyncScanResultsRequest.setConnectTimeout(3000);
        videoAsyncScanResultsRequest.setReadTimeout(6000);
        try {
            HttpResponse httpResponse = client.doAction(videoAsyncScanResultsRequest);
            if (httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttpContent(), "UTF-8"));
                System.out.println(JSON.toJSONString(scrResponse, true));
                int requestCode = scrResponse.getIntValue("code");
                // 每一个视频的检测结果。
                JSONArray taskResults = scrResponse.getJSONArray("data");
                if (200 == requestCode) {
                    for (Object taskResult : taskResults) {
                        // 单个视频的处理结果。
                        int taskCode = ((JSONObject) taskResult).getIntValue("code");
                        if (200 == taskCode) {
                            // taskId。
                            System.out.println(((JSONObject) taskResult).getString("taskId"));
                            // 检测结果。
                            JSONArray results = ((JSONObject) taskResult).getJSONArray("results");
                            for (Object result : results) {
                                // 视频检测结果的分类。
                                System.out.println(((JSONObject) result).getString("label"));
                                // 置信度分数，取值范围：0（表示置信度最低）~100（表示置信度最高）。
                                System.out.println(((JSONObject) result).getString("rate"))
```

```
;
                                    // 视频检测场景，与调用请求中的检测场景一致。
                                    System.out.println(((JSONObject) result).getString("scene")
);
                                    // 建议您执行的后续操作。取值：
                                    // pass：结果正常，无需进行其他操作。
                                    // review：结果不确定，需要人工审核。
                                    // block：结果违规，建议直接删除或者限制公开。
                                    System.out.println(((JSONObject) result).getString("suggest
ion"));
                                }
                            } else {
                                // 单个视频处理失败，原因视具体的情况详细分析。
                                System.out.println("task process fail. task response:" + JSON.t
oJSONString(taskResult));
                            }
                        }
                    } else {
                        /**
                         * 表明请求整体处理失败，原因视具体的情况详细分析。
                         */
                        System.out.println("the whole scan request failed. response:" + JSON.to
JSONString(scrResponse));
                    }
                } else {
                    System.out.println("response not success. status:" + httpResponse.getStatus
());
                }
            } catch (ServerException e) {
                e.printStackTrace();
            } catch (ClientException e) {
                e.printStackTrace();
            }
        }
}
```

提交视频同步检测任务

接口	描述	支持的地域
VideoSyncScanRequest	提交视频同步检测任务，同步检测视频中的风险内容。 **说明** 同步检测只支持传递视频帧序列，不支持检测视频文件，推荐使用异步检测接口。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

说明 以下代码中使用帧序列的方式提交待检测的视频。

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.green.model.v20180509.VideoSyncScanRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.LinkedHashMap;
import java.util.List;
import java.util.Map;
import java.util.UUID;
public class Main {
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile.getProfile("cn-shanghai", "请填写您的AccessKey ID", "请填写您的AccessKey Secret");
        DefaultProfile.addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        VideoSyncScanRequest videoSyncScanRequest = new VideoSyncScanRequest();
        videoSyncScanRequest.setAcceptFormat(FormatType.JSON); // 指定API返回格式。
        videoSyncScanRequest.setMethod(com.aliyuncs.http.MethodType.POST); // 指定请求方法。
        List<Map<String, Object>> tasks = new ArrayList<Map<String, Object>>();
        Map<String, Object> task = new LinkedHashMap<String, Object>();
        task.put("dataId", UUID.randomUUID().toString());
        List<Map<String, Object>> frames = new ArrayList<Map<String, Object>>();
        Map<String, Object> frame1 = new LinkedHashMap<String, Object>();
        frame1.put("offset", 0);
        frame1.put("url", "视频截帧地址1");
        Map<String, Object> frame2 = new LinkedHashMap<String, Object>();
        frame2.put("offset", 5);
        frame2.put("url", "视频截帧地址2");
        Map<String, Object> frame3 = new LinkedHashMap<String, Object>();
        frame3.put("offset", 10);
        frame3.put("url", "视频截帧地址3");
        frames.addAll(Arrays.asList(frame1, frame2, frame3));
        task.put("frames", frames);
        tasks.add(task);
        /**
         * 设置要检测的场景。计费依据此处传递的场景计算。
         * 视频默认1秒截取一帧，您可以自行控制截帧频率，收费按照视频的截帧数量以及每一帧的检测场景进行计算。
         * 举例：1分钟的视频截帧60张，检测色情（对应场景参数porn）和暴恐涉政（对应场景参数terrorism）2个场景，收费按照60张色情+60张暴恐涉政进行计算。
         */
```

```
        JSONObject data = new JSONObject();
        data.put("scenes", Arrays.asList("porn", "terrorism"));
        data.put("tasks", tasks);
        videoSyncScanRequest.setHttpContent(data.toJSONString().getBytes("UTF-8"), "UTF-8",
FormatType.JSON);
        /**
         * 请务必设置超时时间。
         */
        videoSyncScanRequest.setConnectTimeout(3000);
        videoSyncScanRequest.setReadTimeout(10000);
        try {
            HttpResponse httpResponse = client.doAction(videoSyncScanRequest);
            if (httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttpCo
ntent(), "UTF-8"));
                System.out.println(JSON.toJSONString(scrResponse, true));
                int requestCode = scrResponse.getIntValue("code");
                // 每一个视频的检测结果。
                JSONArray taskResults = scrResponse.getJSONArray("data");
                if (200 == requestCode) {
                    for (Object taskResult : taskResults) {
                        // 单个视频的处理结果。
                        int taskCode = ((JSONObject) taskResult).getIntValue("code");
                        if (200 == taskCode) {
                            // taskId。
                            System.out.println(((JSONObject) taskResult).getString("taskId"
));
                            // 检测结果。
                            JSONArray results = scrResponse.getJSONArray("results");
                            for (Object result : results) {
                                // 视频检测结果的分类。
                                System.out.println(((JSONObject) result).getString("label")
);
                                // 置信度分数，取值范围：0（表示置信度最低）~100（表示置信度最高）
。
                                System.out.println(((JSONObject) result).getString("rate"))
;
                                // 视频检测场景，和调用请求中的场景对应。
                                System.out.println(((JSONObject) result).getString("scene")
);
                                // 建议您执行的后续操作。取值：
                                // pass：结果正常，无需进行其余操作。
                                // review：结果不确定，需要进行人工审核。
                                // block：结果违规，建议直接删除或者限制公开。
                                System.out.println(((JSONObject) result).getString("suggest
ion"));
                            }
                        } else {
                            // 单个视频处理失败，原因视具体的情况详细分析。
                            System.out.println("task process fail. task response:" + JSON.t
oJSONString(taskResult));
                        }
                    }
                } else {
```

```
                    /**
                     * 表明请求整体处理失败，原因视具体的情况详细分析。
                     */
                    System.out.println("the whole scan request failed. response:" + JSON.to
JSONString(scrResponse));
                }
            } else {
                System.out.println("response not success. status:" + httpResponse.getStatus
());
            }
        } catch (ServerException e) {
            e.printStackTrace();
        } catch (ClientException e) {
            e.printStackTrace();
        }
    }
}
```

视频检测结果反馈

如果您认为视频检测结果与您的预期不符，可以通过视频检测结果反馈接口，对检测结果进行纠正（系统会根据您反馈的结果，将视频截帧添加到相似图片的黑名单库或者白名单库）。当您再次提交相似的内容进行检测时，以您反馈的label返回结果。

关于接口的说明，请参见检测结果反馈。

接口	描述	支持的Region
VideoFeedbackRequest	提交视频检测结果的反馈，以人工反馈的检测结果纠正算法检测结果。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.green.model.v20180509.VideoFeedbackRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.http.MethodType;
import com.aliyuncs.http.ProtocolType;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import java.util.ArrayList;
import java.util.Arrays;
import java.util.List;
public class VideoFeedbackSample {
    public static void main(String[] args) throws Exception {
```

```
        IClientProfile profile = DefaultProfile
                .getProfile("cn-shanghai", "请填写您的AccessKey ID", "请填写您的AccessKey Secr
et");
        DefaultProfile
                .addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        VideoFeedbackRequest videoFeedbackRequest = new VideoFeedbackRequest();
        // 指定API返回格式。
        videoFeedbackRequest.setAcceptFormat(FormatType.JSON);
        // 指定请求方法。
        videoFeedbackRequest.setMethod(MethodType.POST);
        videoFeedbackRequest.setEncoding("utf-8");
        // 支持HTTP和HTTPS。
        videoFeedbackRequest.setProtocol(ProtocolType.HTTP);
        List<JSONObject> framesList = new ArrayList();
        JSONObject frame1 = new JSONObject();
        frame1.put("url", "视频截帧链接_1");
        frame1.put("offset", "100");
        framesList.add(frame1);
        // scenes：检测场景，支持指定多个场景。
        // suggestion：期望的检测结果，pass：正常；block：违规。
        JSONObject httpBody = new JSONObject();
        httpBody.put("dataId", "检测数据ID");
        httpBody.put("taskId", "视频审核任务ID");
        httpBody.put("url", "视频链接");
        httpBody.put("suggestion", "block");
        httpBody.put("frames", framesList);
        httpBody.put("scenes", Arrays.asList("ad", "terrorism"));
        httpBody.put("note", "备注信息");
        videoFeedbackRequest.setHttpContent(org.apache.commons.codec.binary.StringUtils.get
BytesUtf8(httpBody.toJSONString()),
                "UTF-8", FormatType.JSON);
        videoFeedbackRequest.setConnectTimeout(3000);
        videoFeedbackRequest.setReadTimeout(10000);
        try {
            HttpResponse httpResponse = client.doAction(videoFeedbackRequest);
            if (httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttpCo
ntent(), "UTF-8"));
                System.out.println(JSON.toJSONString(scrResponse, true));
                int requestCode = scrResponse.getIntValue("code");
                if (200 == requestCode) {
                    // 调用成功
                } else {
                    /**
                     * 表明请求整体处理失败，原因视具体的情况详细分析。
                     */
                    System.out.println("the whole request failed. response:" + JSON.toJSONS
tring(scrResponse));
                }
            } else {
                System.out.println("response not success. status:" + httpResponse.getStatus
());
            }
```

```
        } catch (ServerException e) {
            e.printStackTrace();
        } catch (ClientException e) {
            e.printStackTrace();
        }
    }
}
```

2.3.3. 文本反垃圾检测

本文介绍了如何使用Java SDK文本反垃圾接口，对文本内容进行色情、暴恐、涉政等风险进行识别。

功能描述

文本反垃圾接口目前仅支持同步检测。关于参数的详细说明，请参见文本同步检测。

一次请求可以检测多条文本，也可以检测单条文本。按实际检测的文本条数进行计费，请参见计费方式概述。

前提条件

- 安装Java依赖。关于安装Java依赖的具体操作，请参见安装Java依赖。

  > **说明** 请一定按照安装Java依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

文本内容检测

文本垃圾检测支持自定义关键词，例如添加一些竞品关键词等。如果被检测的文本中包含您添加的关键词，算法会返回您suggestion为block。

您可以前往内容安全控制台添加关键词，也可以通过API接口添加关键词。

接口	描述	支持的Region
TextScanRequest	提交文本反垃圾检测任务，检测场景参数请传递antispam（scenes=antispam）。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyun.oss.ClientException;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.green.model.v20180509.TextScanRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.profile.DefaultProfile;
```

```
import com.aliyuncs.profile.IClientProfile;
import java.util.*;
public class Main {
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile
            .getProfile("cn-shanghai", "您的AccessKey ID", "您的AccessKey Secret");
        DefaultProfile
            .addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        TextScanRequest textScanRequest = new TextScanRequest();
        textScanRequest.setAcceptFormat(FormatType.JSON); // 指定API返回格式。
        textScanRequest.setHttpContentType(FormatType.JSON);
        textScanRequest.setMethod(com.aliyuncs.http.MethodType.POST); // 指定请求方法。
        textScanRequest.setEncoding("UTF-8");
        textScanRequest.setRegionId("cn-shanghai");
        List<Map<String, Object>> tasks = new ArrayList<Map<String, Object>>();
        Map<String, Object> task1 = new LinkedHashMap<String, Object>();
        task1.put("dataId", UUID.randomUUID().toString());
        /**
         * 待检测的文本，长度不超过10000个字符。
         */
        task1.put("content", "test content");
        tasks.add(task1);
        JSONObject data = new JSONObject();
        /**
         * 检测场景。文本垃圾检测请传递antispam。
         **/
        data.put("scenes", Arrays.asList("antispam"));
        data.put("tasks", tasks);
        System.out.println(JSON.toJSONString(data, true));
        textScanRequest.setHttpContent(data.toJSONString().getBytes("UTF-8"), "UTF-8", Form
atType.JSON);
        // 请务必设置超时时间。
        textScanRequest.setConnectTimeout(3000);
        textScanRequest.setReadTimeout(6000);
        try {
            HttpResponse httpResponse = client.doAction(textScanRequest);
            if(httpResponse.isSuccess()){
                JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttpCo
ntent(), "UTF-8"));
                System.out.println(JSON.toJSONString(scrResponse, true));
                if (200 == scrResponse.getInteger("code")) {
                    JSONArray taskResults = scrResponse.getJSONArray("data");
                    for (Object taskResult : taskResults) {
                        if(200 == ((JSONObject)taskResult).getInteger("code")){
                            JSONArray sceneResults = ((JSONObject)taskResult).getJSONArray(
"results");
                            for (Object sceneResult : sceneResults) {
                                String scene = ((JSONObject)sceneResult).getString("scene")
;
                                String suggestion = ((JSONObject)sceneResult).getString("su
ggestion");
                                // 根据scene和suggetion做相关处理。
                                // suggestion为pass表示未命中垃圾。suggestion为block表示命中了
```

垃圾，可以通过label字段查看命中的垃圾分类。

```
                                System.out.println("args = [" + scene + "]");
                                System.out.println("args = [" + suggestion + "]");
                            }
                        }else{
                            System.out.println("task process fail:" + ((JSONObject)taskResu
lt).getInteger("code"));
                        }
                    }
                } else {
                    System.out.println("detect not success. code:" + scrResponse.getInteger
("code"));
                }
            }else{
                System.out.println("response not success. status:" + httpResponse.getStatus
());
            }
        } catch (ServerException e) {
            e.printStackTrace();
        } catch (ClientException e) {
            e.printStackTrace();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

文本反垃圾结果反馈

如果您认为我们的文本检测结果与您的期望不符，您可以通过文本反垃圾结果反馈接口纠正算法检测结果。

系统会将您反馈的结果添加进相似文本黑库或相似文本白库，在您下次提交相似的内容进行检测时，以您通过反馈接口传递的**label**作为结果返回。关于接口的说明，请参见文本反垃圾结果反馈API文档。

接口	描述	支持的地域
TextFeedbackRequest	提交文本反垃圾检测结果的反馈，以人工反馈的检测结果纠正算法检测结果。	• **cn-shanghai**：华东2（上海） • **cn-beijing**：华北2（北京） • **cn-shenzhen**：华南1（深圳） • **ap-southeast-1**：新加坡

示例代码

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.green.model.v20180509.TextFeedbackRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.http.MethodType;
```

```
import com.aliyuncs.http.ProtocolType;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
public class TextFeedbackSample {
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile
                .getProfile("cn-shanghai", "请填写您的AccessKey ID", "请填写您的AccessKey Secr
et");
        DefaultProfile
                .addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        TextFeedbackRequest textFeedbackRequest = new TextFeedbackRequest();
        // 指定API返回格式。
        textFeedbackRequest.setAcceptFormat(FormatType.JSON);
        // 指定请求方法。
        textFeedbackRequest.setMethod(MethodType.POST);
        textFeedbackRequest.setEncoding("utf-8");
        // 支持HTTP和HTTPS。
        textFeedbackRequest.setProtocol(ProtocolType.HTTP);
        // label：反馈的分类，与具体的检测场景对应。
        JSONObject httpBody = new JSONObject();
        httpBody.put("dataId", "检测数据ID");
        httpBody.put("taskId", "文本审核任务ID");
        httpBody.put("content", "文本内容");
        httpBody.put("label", "spam");
        httpBody.put("note", "备注信息");
        textFeedbackRequest.setHttpContent(org.apache.commons.codec.binary.StringUtils.getB
ytesUtf8(httpBody.toJSONString()),
                "UTF-8", FormatType.JSON);
        textFeedbackRequest.setConnectTimeout(3000);
        textFeedbackRequest.setReadTimeout(10000);
        try {
            HttpResponse httpResponse = client.doAction(textFeedbackRequest);
            if (httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttpCo
ntent(), "UTF-8"));
                System.out.println(JSON.toJSONString(scrResponse, true));
                int requestCode = scrResponse.getIntValue("code");
                if (200 == requestCode) {
                    // 调用成功。
                } else {
                    /**
                     * 表明请求整体处理失败，原因视具体的情况详细分析。
                     */
                    System.out.println("the whole scan request failed. response:" + JSON.to
JSONString(scrResponse));
                }
            } else {
                System.out.println("response not success. status:" + httpResponse.getStatus
());
            }
        } catch (ServerException e) {
            e.printStackTrace();
        } catch (ClientException e) {
```

```
            e.printStackTrace();
        }
    }
}
```

2.3.4. 语音反垃圾检测

本文介绍了如何使用Java SDK语音反垃圾接口，检测实时语音流或语音文件中的垃圾内容。

功能描述

语音流检测和语音文件检测均为异步检测，检测结果需要您以轮询或者回调的方式获取。关于调用请求中的检测场景参数scenes，返回结果中的分类参数label，以及操作建议参数suggestion的说明，请参见语音异步检测。

语音检测按照检测的语音文件、语音流的时间长度进行计费，计费粒度为分钟，每天累计检测总时长进行计量统计，每天检测总时长不足一分钟的按照一分钟进行计费。

前提条件

- 安装Java依赖。关于安装Java依赖的具体操作，请参见安装Java依赖。

  > **说明** 请一定按照安装Java依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

提交语音异步检测任务

接口	描述	支持的地域
VoiceAsyncScanRequest	异步检测语音流或语音文件中是否包含违规内容。语音流格式支持： • RTMP • HTTP • FLV	**cn-shanghai**：华东2（上海）

示例代码

- 提交语音URL进行检测

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.green.model.v20180509.VoiceAsyncScanRequest;
import com.aliyuncs.green.model.v20180509.VoiceAsyncScanResultsRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import java.util.*;
public class Main {
```

```
public class Main {
    public static void main(String[] args) throws Exception {
        // 请替换成您的AccessKey ID、AccessKey Secret。
        IClientProfile profile = DefaultProfile.getProfile("cn-shanghai", "您的AccessKey ID", "您的AccessKey Secret");
        final IAcsClient client = new DefaultAcsClient(profile);
        VoiceAsyncScanRequest asyncScanRequest = new VoiceAsyncScanRequest(); //class different vs common
        asyncScanRequest.setAcceptFormat(FormatType.JSON); // 指定API返回格式。
        asyncScanRequest.setMethod(com.aliyuncs.http.MethodType.POST); // 指定请求方法。
        asyncScanRequest.setRegionId("cn-shanghai");
        asyncScanRequest.setConnectTimeout(3000);
        asyncScanRequest.setReadTimeout(6000);
        List<Map<String, Object>> tasks = new ArrayList<Map<String, Object>>();
        Map<String, Object> task1 = new LinkedHashMap<String, Object>();
        // 请将下面的地址修改为要检测的语音文件的地址。
        task1.put("url", "https://xxxxxxxxxxxxx");
        tasks.add(task1);
        JSONObject data = new JSONObject();
        System.out.println("==========Task count:" + tasks.size());
        data.put("scenes", Arrays.asList("antispam"));
        data.put("tasks", tasks);
        // 如果是语音流检测，则修改为true。
        data.put("live", false);
        asyncScanRequest.setHttpContent(data.toJSONString().getBytes("UTF-8"), "UTF-8", FormatType.JSON);
        System.out.println(JSON.toJSONString(data, true));
        try {
            HttpResponse httpResponse = client.doAction(asyncScanRequest);
            if (httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttpContent(), "UTF-8"));
                System.out.println(JSON.toJSONString(scrResponse, true));
                if (200 == scrResponse.getInteger("code")) {
                    JSONArray taskResults = scrResponse.getJSONArray("data");
                    for (Object taskResult : taskResults) {
                        Integer code = ((JSONObject) taskResult).getInteger("code");
                        if (200 == code) {
                            final String taskId = ((JSONObject) taskResult).getString("taskId");
                            System.out.println("submit async task success, taskId = [" + taskId + "]");
                        } else {
                            System.out.println("task process fail: " + code);
                        }
                    }
                } else {
                    System.out.println("detect not success. code: " + scrResponse.getInteger("code"));
                }
            } else {
                System.out.println("response not success. status: " + httpResponse.getStatus());
            }
        } catch (Exception e) {
```

```
            e.printStackTrace();
        }
    }
}
```

- 提交本地语音文件进行检测

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.green.model.v20180509.VoiceAsyncScanRequest;
import com.aliyuncs.green.model.v20180509.VoiceAsyncScanResultsRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import com.aliyuncs.green.extension.uploader.ClientUploader;
import java.util.*;
public class Main {
    public static void main(String[] args) throws Exception {
        // 请替换成您的AccessKey ID、AccessKey Secret。
        IClientProfile profile = DefaultProfile.getProfile("cn-shanghai", "您的AccessKey ID", "您的AccessKey Secret");
        final IAcsClient client = new DefaultAcsClient(profile);
        VoiceAsyncScanRequest asyncScanRequest = new VoiceAsyncScanRequest(); //class different vs common
        asyncScanRequest.setAcceptFormat(FormatType.JSON); // 指定API返回格式。
        asyncScanRequest.setMethod(com.aliyuncs.http.MethodType.POST); // 指定请求方法。
        asyncScanRequest.setRegionId("cn-shanghai");
        asyncScanRequest.setConnectTimeout(3000);
        asyncScanRequest.setReadTimeout(6000);
        List<Map<String, Object>> tasks = new ArrayList<Map<String, Object>>();
        Map<String, Object> task1 = new LinkedHashMap<String, Object>();
        task1.put("type", "file");
        String url = null;
        ClientUploader uploader = ClientUploader.getVideoClientUploader(profile, false);
        try{
            url = uploader.uploadFile("d:/test.mp4");
        }catch (Exception e){
            e.printStackTrace();
        }
        task1.put("url", url);
        tasks.add(task1);
        JSONObject data = new JSONObject();
        System.out.println("==========Task count:" + tasks.size());
        data.put("scenes", Arrays.asList("antispam"));
        data.put("tasks", tasks);
        asyncScanRequest.setHttpContent(data.toJSONString().getBytes("UTF-8"), "UTF-8", FormatType.JSON);
        System.out.println(JSON.toJSONString(data, true));
        try {
            HttpResponse httpResponse = client.doAction(asyncScanRequest);
            if (httpResponse.isSuccess()) {
```

```
            if (httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttp
Content(), "UTF-8"));
                System.out.println(JSON.toJSONString(scrResponse, true));
                if (200 == scrResponse.getInteger("code")) {
                    JSONArray taskResults = scrResponse.getJSONArray("data");
                    for (Object taskResult : taskResults) {
                        Integer code = ((JSONObject) taskResult).getInteger("code");
                        if (200 == code) {
                            final String taskId = ((JSONObject) taskResult).getString("ta
skId");
                            System.out.println("submit async task success, taskId = [" +
taskId + "]");
                        } else {
                            System.out.println("task process fail: " + code);
                        }
                    }
                } else {
                    System.out.println("detect not success. code: " + scrResponse.getInte
ger("code"));
                }
            } else {
                System.out.println("response not success. status: " + httpResponse.getSta
tus());
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

查询语音异步检测结果

接口	描述	支持的地域
VoiceAsyncScanResultsRequest	查询异步语音检测结果。	`cn-shanghai`：华东2（上海）

示例代码

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.green.model.v20180509.VoiceAsyncScanResultsRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import java.util.*;
public class Main {
    public static void main(String[] args) throws Exception {
        // 请替换成您的AccessKey ID、AccessKey Secret。
        IClientProfile profile = DefaultProfile
```

```
            .getProfile("cn-shanghai", "您的AccessKey ID", "您的AccessKey Secret");
        final IAcsClient client = new DefaultAcsClient(profile);
        pollingScanResult(client, "提交语音异步检测任务后返回的taskId");
    }
    public static void pollingScanResult(IAcsClient client, String taskId) throws Interrupt
edException {
        int failCount = 0;
        boolean stop = false;
        do {
            // 设置每10秒查询一次。
            Thread.sleep(10 * 1000);
            JSONObject scanResult = getScanResult(client, taskId);
            if (scanResult == null || 200 != scanResult.getInteger("code")) {
                failCount++;
                System.out.println(taskId + ": get result fail, failCount=" + failCount);
                if (scanResult != null) {
                    System.out.println(taskId + ": errorMsg:" + scanResult.getString("msg")
);
                }
                if (failCount > 20) {
                    break;
                }
                continue;
            }
            JSONArray taskResults = scanResult.getJSONArray("data");
            if (taskResults.isEmpty()) {
                System.out.println("failed");
                break;
            }
            for (Object taskResult : taskResults) {
                JSONObject result = (JSONObject) taskResult;
                Integer code = result.getInteger("code");
                if (280 == code) {
                    System.out.println(taskId + ": processing status: " + result.getString(
"msg"));
                } else if (200 == code) {
                    System.out.println(taskId + ": ========== SUCCESS ==========");
                    System.out.println(JSON.toJSONString(scanResult, true));
                    System.out.println(taskId + ": ========== SUCCESS ==========");
                    stop = true;
                } else {
                    System.out.println(taskId + ": ========== FAILED ==========");
                    System.out.println(JSON.toJSONString(scanResult, true));
                    System.out.println(taskId + ": ========== FAILED ==========");
                    stop = true;
                }
            }
        } while (!stop);
    }
    private static JSONObject getScanResult(IAcsClient client, String taskId) {
        VoiceAsyncScanResultsRequest getResultsRequest = new VoiceAsyncScanResultsRequest()
;
        getResultsRequest.setAcceptFormat(FormatType.JSON); // 指定API返回格式。
        getResultsRequest.setMethod(com.aliyuncs.http.MethodType.POST); // 指定请求方法。
```

```
        getResultsRequest.setEncoding("utf-8");
        getResultsRequest.setRegionId("cn-shanghai");
        List<Map<String, Object>> tasks = new ArrayList<Map<String, Object>>();
        Map<String, Object> task1 = new LinkedHashMap<String, Object>();
        task1.put("taskId", taskId);
        tasks.add(task1);
        /**
         * 请务必设置超时时间。
         */
        getResultsRequest.setConnectTimeout(3000);
        getResultsRequest.setReadTimeout(6000);
        try {
            getResultsRequest.setHttpContent(JSON.toJSONString(tasks).getBytes("UTF-8"), "UTF-8", FormatType.JSON);
            HttpResponse httpResponse = client.doAction(getResultsRequest);
            if (httpResponse.isSuccess()) {
                return JSON.parseObject(new String(httpResponse.getHttpContent(), "UTF-8"));
            } else {
                System.out.println("response not success. status: " + httpResponse.getStatus());
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
        return null;
    }
}
```

短语音同步检测

示例代码

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.green.model.v20180509.VoiceSyncScanRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import java.util.*;
public class Main {
    public static void main(String[] args) throws Exception {
        // 请替换成您的AccessKey ID、AccessKey Secret。
        IClientProfile profile = DefaultProfile.getProfile("cn-shanghai", "您的AccessKey ID", "您的AccessKey Secret");
        final IAcsClient client = new DefaultAcsClient(profile);
        VoiceSyncScanRequest asyncScanRequest = new VoiceSyncScanRequest();
        asyncScanRequest.setAcceptFormat(FormatType.JSON); // 指定API返回格式。
        asyncScanRequest.setMethod(com.aliyuncs.http.MethodType.POST); // 指定请求方法。
        asyncScanRequest.setRegionId("cn-shanghai");
```

```
        asyncScanRequest.setConnectTimeout(3000);
        // 由于同步语音检测比较耗时，因此建议将超时时间设置在15秒以上。
        asyncScanRequest.setReadTimeout(15000);
        List<Map<String, Object>> tasks = new ArrayList<Map<String, Object>>();
        Map<String, Object> task1 = new LinkedHashMap<String, Object>();
        // 请将下面的地址修改为要检测的语音文件的地址。
        task1.put("url", "https://xxxxxxxxxxxxx");
        tasks.add(task1);
        JSONObject data = new JSONObject();
        System.out.println("==========Task count:" + tasks.size());
        data.put("scenes", Arrays.asList("antispam"));
        data.put("tasks", tasks);
        asyncScanRequest.setHttpContent(data.toJSONString().getBytes("UTF-8"), "UTF-8", For
matType.JSON);
        System.out.println(JSON.toJSONString(data, true));
        try {
            HttpResponse httpResponse = client.doAction(asyncScanRequest);
            if (httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttpCo
ntent(), "UTF-8"));
                System.out.println(JSON.toJSONString(scrResponse, true));
                if (200 == scrResponse.getInteger("code")) {
                    JSONArray taskResults = scrResponse.getJSONArray("data");
                    for (Object taskResult : taskResults) {
                        Integer code = ((JSONObject) taskResult).getInteger("code");
                        if (200 == code) {
                            System.out.println("task process success, result:" + JSON.toJSO
NString(taskResult));
                        } else {
                            System.out.println("task process fail: " + JSON.toJSONString(ta
skResult));
                        }
                    }
                } else {
                    System.out.println("detect not success. code: " + scrResponse.getIntege
r("code"));
                }
            } else {
                System.out.println("response fail:" + new String(httpResponse.getHttpConten
t(), "UTF-8"));
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

2.3.5. 文件审核

本文介绍如何使用Java SDK文件审核接口，检测文件中的文字和图片信息。

功能描述

文件审核目前只支持异步检测（异步检测不会实时返回检测结果）任务。关于参数的详细说明，请参见提交文件检测任务。

- 支持的文件类型：
  - 支持检测以下文件中的文本内容。

    PDF、*WORD*、*TXT*、*PPT*、*EXCEL*、*OUTLOOK*、*VISIO*、*ZIP*、*TAR*、*RTF*。
  - 支持检测*PDF*文件中的图片内容。
- 支持的文件大小：5 MB以内。

前提条件

- 安装Java依赖。关于安装Java依赖的具体操作，请参见安装Java依赖。

  > **说明** 请一定按照安装Java依赖页面中的版本安装，否则会导致调用失败。
- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

提交文件异步检测任务

接口	描述	支持的地域
FileAsyncScanRequest	提交文件异步检测任务，对文件中的文字和图片进行检测。	• **cn-shanghai**：华东2（上海） • **cn-beijing**：华北2（北京）

示例代码

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.green.model.v20180509.FileAsyncScanRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import java.util.Arrays;
public class Main {
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile.getProfile("cn-shanghai", "请填写您的AccessKey ID", "请填写您的AccessKey Secret");
        DefaultProfile
                .addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        FileAsyncScanRequest fileAsyncScanRequest = new FileAsyncScanRequest();
        // 指定API返回格式。
        fileAsyncScanRequest.setAcceptFormat(FormatType.JSON);
        // 指定请求方法。
        fileAsyncScanRequest.setMethod(com.aliyuncs.http.MethodType.POST);
        fileAsyncScanRequest.setEncoding("utf-8");
        // textScenes：检测内容包含文本时，指定检测场景，取值：antispam。imageScenes：检测内容包含图片时，指定检测场景
```

```
图片时，指定检测场景。
        JSONObject data = new JSONObject();
        data.put("textScenes", Arrays.asList("antispam"));
        data.put("imageScenes", Arrays.asList("porn", "ad"));
        JSONObject task = new JSONObject();
        task.put("dataId", "检测数据ID");
        task.put("url", "文件链接");
        data.put("tasks", Arrays.asList(task));
        fileAsyncScanRequest.setHttpContent(org.apache.commons.codec.binary.StringUtils.get
BytesUtf8(data.toJSONString()),
                "UTF-8", FormatType.JSON);
        /**
         * 请设置超时时间，服务端全链路处理超时时间为10秒，请做相应设置。
         * 如果您设置的ReadTimeout小于服务端处理的时间，程序中会获得一个ReadTimeout异常。
         */
        fileAsyncScanRequest.setConnectTimeout(3000);
        fileAsyncScanRequest.setReadTimeout(10000);
        HttpResponse httpResponse = null;
        try {
            httpResponse = client.doAction(fileAsyncScanRequest);
            if (httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttpCo
ntent(), "UTF-8"));
                System.out.println(JSON.toJSONString(scrResponse, true));
                int requestCode = scrResponse.getIntValue("code");
                // 每一个文件的检测结果。
                JSONArray taskResults = scrResponse.getJSONArray("data");
                if (200 == requestCode) {
                    for (Object taskResult : taskResults) {
                        // 单个文件的处理结果。
                        int taskCode = ((JSONObject) taskResult).getIntValue("code");
                        if (200 == taskCode) {
                            // 保存taskId用于轮询结果。
                            System.out.println(((JSONObject) taskResult).getString("taskId"
));
                        } else {
                            // 单个文件处理失败，原因视具体的情况详细分析。
                            System.out.println("task process fail. task response:" + JSON.t
oJSONString(taskResult));
                        }
                    }
                } else {
                    /**
                     * 表明请求整体处理失败，原因视具体的情况详细分析。
                     */
                    System.out.println("the whole scan request failed. response:" + JSON.to
JSONString(scrResponse));
                }
            } else {
                System.out.println("response not success. status:" + httpResponse.getStatus
());
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
```

```
        }
    }
}
```

获取文件异步检测任务结果

接口	描述	支持的Region
FileAsyncScanResultsRequest	获取文件异步检测结果。	• **cn-shanghai**：华东2（上海） • **cn-beijing**：华北2（北京）

示例代码

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.green.model.v20180509.FileAsyncScanResultsRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import java.util.Arrays;
public class Main {
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile.getProfile("cn-shanghai", "请填写您的AccessKey ID", "请填写您的AccessKey Secret");
        DefaultProfile
                .addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        FileAsyncScanResultsRequest fileAsyncScanResultsRequest = new FileAsyncScanResultsRequest();
        // 指定API返回格式。
        fileAsyncScanResultsRequest.setAcceptFormat(FormatType.JSON);
        // 指定请求方法。
        fileAsyncScanResultsRequest.setMethod(com.aliyuncs.http.MethodType.POST);
        fileAsyncScanResultsRequest.setEncoding("utf-8");
        JSONObject data = new JSONObject();
        data.put("taskIds", Arrays.asList("文件审核任务ID"));
        fileAsyncScanResultsRequest.setHttpContent(org.apache.commons.codec.binary.StringUtils.getBytesUtf8(data.toJSONString()),
                "UTF-8", FormatType.JSON);
        /**
         * 请设置超时时间，服务端全链路处理超时时间为10秒，请做相应设置。
         * 如果您设置的ReadTimeout小于服务端处理的时间，程序中会获得一个ReadTimeout异常。
         */
        fileAsyncScanResultsRequest.setConnectTimeout(3000);
        fileAsyncScanResultsRequest.setReadTimeout(10000);
        HttpResponse httpResponse = null;
        try {
            httpResponse = client.doAction(fileAsyncScanResultsRequest);
            if (httpResponse.isSuccess()) {
```

```
                JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttpCo
ntent(), "UTF-8"));
                System.out.println(JSON.toJSONString(scrResponse, true));
                int requestCode = scrResponse.getIntValue("code");
                // 每一个文件的检测结果。
                JSONArray taskResults = scrResponse.getJSONArray("data");
                if (200 == requestCode) {
                    for (Object taskResult : taskResults) {
                        // 单个文件的处理结果。
                        int taskCode = ((JSONObject) taskResult).getIntValue("code");
                        if (200 == taskCode) {
                            // taskId。
                            System.out.println(((JSONObject) taskResult).getString("taskId"
));
                            // 传入textScenes时的文本扫描结果。
                            JSONArray textResults = ((JSONObject) taskResult).getJSONArray(
"textResults");
                            // 传入imageScenes时的图片扫描结果。
                            JSONArray imageResults = ((JSONObject) taskResult).getJSONArray
("imageResults");
                        } else {
                            // 单个视频处理失败，原因视具体的情况详细分析。
                            System.out.println("task process fail. task response:" + JSON.t
oJSONString(taskResult));
                        }
                    }
                } else {
                    /**
                     * 表明请求整体处理失败，原因视具体的情况详细分析。
                     */
                    System.out.println("the whole scan request failed. response:" + JSON.to
JSONString(scrResponse));
                }
            } else {
                System.out.println("response not success. status:" + httpResponse.getStatus
());
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

2.3.6. 网页审核

本文介绍如何使用Java SDK网页审核接口检测网页中的文字和图片信息，并同步返回检测出来的结果。

功能描述

图片审核支持同步检测和异步检测两种方式。

- 同步检测实时返回检测结果。关于参数的详细信息，请参见网页同步检测。
- 异步检测需要您轮询结果或者通过callback回调通知获取检测结果。关于参数的详细信息，请参见网页异

步检测。

前提条件

- 安装Java依赖。关于安装Java依赖的具体操作，请参见安装Java依赖。

  > **说明** 请一定按照安装Java依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

提交网页同步检测任务

接口	描述	支持的地域
WebpageSyncScanRequest	提交网页同步检测任务，对网页中的文字和图片进行检测。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京）

示例代码

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.green.model.v20180509.WebpageSyncScanRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.http.MethodType;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import java.util.Arrays;
import java.util.UUID;
public class WebPageSyncScanSample {
    private static final String aliyunAccessKeyId = "您的AccessKey ID";
    private static final String aliyunAccessKeySecret = "您的AccessKey Secret";
    private static String regionId = "cn-shanghai";
    private static String endpoint = "green.cn-shanghai.aliyuncs.com";
    public static void main(String[] args) {
        IClientProfile profile = DefaultProfile.getProfile(regionId, aliyunAccessKeyId, ali
yunAccessKeySecret);
        DefaultProfile.addEndpoint(regionId, "Green", endpoint);
        IAcsClient client = new DefaultAcsClient(profile);
        WebpageSyncScanRequest request = new WebpageSyncScanRequest();
        // 指定返回格式。
        request.setAcceptFormat(FormatType.JSON);
        // 指定请求方法。
        request.setMethod(MethodType.POST);
        // 请设置连接超时时间，您可根据实际情况进行修改。单位：ms。
        request.setConnectTimeout(3000);
        // 请设置超时时间，您可根据实际情况进行修改。单位：ms。
        request.setReadTimeout(9000);
        // 构建请求消息。
        JSONObject requestContent = new JSONObject();
```

```
        requestContent.put("bizType", "webPage");
        requestContent.put("textScenes", Arrays.asList("antispam"));
        requestContent.put("imageScenes", Arrays.asList("porn", "terrorism"));
        JSONObject task = new JSONObject();
        task.put("dataId", UUID.randomUUID().toString());
        task.put("url", "https://example.com");  // 待检测的网页URL。
        requestContent.put("tasks", Arrays.asList(task));
        try {
            request.setHttpContent(requestContent.toJSONString().getBytes("UTF-8"), "UTF-8"
, FormatType.JSON);
            HttpResponse httpResponse = client.doAction(request);
            String responseBody = httpResponse.getHttpContentString();
            // HTTP请求成功。
            if (httpResponse.isSuccess()) {
                JSONObject responseData = JSON.parseObject(responseBody);
                int respCode = responseData.getIntValue("code");
                // 结果处理。
                if (respCode == 200) {
                    JSONArray taskResults = responseData.getJSONArray("data");
                    for (int i = 0; i < taskResults.size(); i++) {
                        JSONObject taskRst = taskResults.getJSONObject(i);
                        int taskCode = taskRst.getIntValue("code");
                        if (taskCode == 200) {
                            // 任务处理成功。
                            System.out.println("task success : " + JSON.toJSONString(taskRs
t));

                            // 根据taskRst中返回的检测结果进行业务处理。
                        } else {
                            // 单个任务失败，分析失败原因。
                            System.out.println("task fail : " + JSON.toJSONString(taskRst))
;

                        }
                    }
                } else {
                    // 整个结果失败，需要分析失败原因。
                    System.out.println("response fail : " + responseBody);
                }
            } else {
                // 请求失败，需要分析错误内容。
                System.out.println("request fail : " + responseBody);
            }
        } catch (Exception e) {
            // 请求发生异常的处理。
            e.printStackTrace();
        }
    }
}
```

提交网页异步检测任务

接口	描述	支持的Region
WebpageAsyncScanRequest	提交网页异步检测任务，对网页中的文字和图片进行检测。	• **cn-shanghai**：华东2（上海） • **cn-beijing**：华北2（北京）

示例代码

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.green.model.v20180509.WebpageAsyncScanRequest;
import com.aliyuncs.green.model.v20180509.WebpageAsyncScanResultsRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.http.MethodType;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import java.util.Arrays;
import java.util.List;
import java.util.UUID;
public class WebAsyncScanSample {
    private static final String aliyunAccessKeyId = "您的AccessKey ID";
    private static final String aliyunAccessKeySecret = "您的AccessKey Secret";
    private static String regionId = "cn-shanghai";
    private static String endpoint = "green.cn-shanghai.aliyuncs.com";
    public static void main(String[] args) throws InterruptedException {
        IClientProfile profile = DefaultProfile.getProfile(regionId, aliyunAccessKeyId, ali
yunAccessKeySecret);
        DefaultProfile.addEndpoint(regionId, "Green", endpoint);
        IAcsClient client = new DefaultAcsClient(profile);
        // 提交网页异步检测任务。
        String taskId = submitAsyncTask(client);
        if (taskId != null) {
            // 轮询任务结果。
            for (int i = 0; i < 100; i++) {
                JSONObject ret = queryResult(client, taskId);
                if (ret == null) {
                    Thread.sleep(5000);
                } else {
                    break;
                }
            }
        }
    }
    private static String submitAsyncTask(IAcsClient client) {
        WebpageAsyncScanRequest request = new WebpageAsyncScanRequest();
        // 指定返回格式。
        request.setAcceptFormat(FormatType.JSON);
        // 指定请求方法。
        request.setMethod(MethodType.POST);
        request.setConnectTimeout(3000);
```

```
        request.setReadTimeout(9000);
        // 构建请求消息。
        JSONObject requestContent = new JSONObject();
        requestContent.put("bizType", "webPage");
        requestContent.put("textScenes", Arrays.asList("antispam"));
        requestContent.put("imageScenes", Arrays.asList("porn"));
        // 通过回调接收结果。
        //requestContent.put("callback", "http://example.com/result/recv");
        //requestContent.put("seed", "seedxxxxxxx");
        JSONObject task = new JSONObject();
        task.put("dataId", UUID.randomUUID().toString());
        // 待检测的网页URL。
        task.put("url", "https://example.com/index.html");
        requestContent.put("tasks", Arrays.asList(task));
        try {
            request.setHttpContent(requestContent.toJSONString().getBytes("UTF-8"), "UTF-8"
, FormatType.JSON);
            HttpResponse httpResponse = client.doAction(request);
            String responseBody = httpResponse.getHttpContentString();
            if (!httpResponse.isSuccess()) {
                // 请求失败，需要分析错误内容。
                System.out.println("request fail : " + responseBody);
            } else {
                JSONObject responseData = JSON.parseObject(responseBody);
                int respCode = responseData.getIntValue("code");
                if (respCode != 200) {
                    System.out.println("response fail : " + responseBody);
                } else {
                    JSONArray taskResults = responseData.getJSONArray("data");
                    JSONObject taskRst = taskResults.getJSONObject(0);
                    int taskCode = taskRst.getIntValue("code");
                    if (taskCode != 200) {
                        System.out.println("task fail : " + JSON.toJSONString(taskRst));
                    } else {
                        System.out.println("task success : " + JSON.toJSONString(taskRst));
                        return taskRst.getString("taskId");
                    }
                }
            }
        } catch (Exception e) {
            // 请求发生异常的处理。
            e.printStackTrace();
        }
        return null;
    }
    private static JSONObject queryResult(IAcsClient client, String taskId) {
        WebpageAsyncScanResultsRequest request = new WebpageAsyncScanResultsRequest();
        // 指定返回格式。
        request.setAcceptFormat(FormatType.JSON);
        // 指定请求方法。
        request.setMethod(MethodType.POST);
        // 请设置连接超时时间，您可根据实际情况进行修改。单位：ms。
        request.setConnectTimeout(3000);
        // 请设置超时时间，您可根据实际情况进行修改。单位：ms。
```

```
        request.setReadTimeout(9000);
        // 构建请求消息。
        List<String> taskIds = Arrays.asList(taskId);
        try {
            request.setHttpContent(JSONObject.toJSONString(taskIds).getBytes("UTF-8"), "UTF
-8", FormatType.JSON);
            HttpResponse httpResponse = client.doAction(request);
            String responseBody = httpResponse.getHttpContentString();
            JSONObject responseData = JSON.parseObject(responseBody);
            if (!httpResponse.isSuccess()) {
                // 请求失败，返回对应的错误码。
                System.out.println("request fail : " + responseBody);
                return responseData;
            } else {
                int respCode = responseData.getIntValue("code");
                if (respCode != 200) {
                    System.out.println("response fail : " + responseBody);
                    return responseData;
                } else {
                    JSONArray taskResults = responseData.getJSONArray("data");
                    JSONObject taskRst = taskResults.getJSONObject(0);
                    int taskCode = taskRst.getIntValue("code");
                    if (taskCode == 280) {
                        System.out.println("task processing : " + JSON.toJSONString(taskRst
));

                        return null;
                    } else if (taskCode != 200) {
                        System.out.println("task fail : " + JSON.toJSONString(taskRst));
                        return taskRst;
                    } else {
                        System.out.println("task success : " + JSON.toJSONString(taskRst));
                        return taskRst;
                    }
                }
            }
        } catch (Exception e) {
            // 请求发生异常的处理。
            e.printStackTrace();
        }
        return null;
    }
}
```

2.4. 人工审核

2.4.1. 图片人工审核

本文介绍如何使用Java SDK图片人工审核接口。

功能描述

如果您认为图片检测结果（机审）与预期不符，可以使用图片人工审核接口。关于参数的详细信息，请参见图片人工审核。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Java依赖。关于安装Java依赖的具体操作，请参见安装Java依赖。

  > **说明** 请一定按照安装Java依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

提交图片人工审核任务

示例代码

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.green.model.v20180509.ImageAsyncManualScanRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.http.MethodType;
import com.aliyuncs.http.ProtocolType;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import java.util.Arrays;
import java.util.UUID;
public class ImageAsyncManualScanSample {
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile.getProfile("cn-shanghai", "您的AccessKey ID"
, "您的AccessKey Secret");
        DefaultProfile.addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com"
);
        IAcsClient client = new DefaultAcsClient(profile);
        ImageAsyncManualScanRequest imageAsyncManualScanRequest = new ImageAsyncManualScanR
equest();
        // 指定API返回格式。
        imageAsyncManualScanRequest.setAcceptFormat(FormatType.JSON);
        // 指定请求方法。
        imageAsyncManualScanRequest.setMethod(MethodType.POST);
        imageAsyncManualScanRequest.setEncoding("utf-8");
        // 支持HTTP和HTTPS。
        imageAsyncManualScanRequest.setProtocol(ProtocolType.HTTP);
        JSONObject task = new JSONObject();
        task.put("dataId", UUID.randomUUID().toString());
        // 设置图片链接。
        task.put("url", "待检测图片链接");
        JSONObject httpBody = new JSONObject();
```

```
        httpBody.put("tasks", Arrays.asList(task));
        // callback、seed用于回调通知，可选参数.
        httpBody.put("callback", "回调地址");
        httpBody.put("seed", "随机字符串");
        imageAsyncManualScanRequest.setHttpContent(org.apache.commons.codec.binary.StringUt
ils.getBytesUtf8(httpBody.toJSONString()),
                "UTF-8", FormatType.JSON);
        /**
         * 请设置超时时间。服务端全链路处理超时时间为10秒，请做相应设置。
         * 如果您设置的ReadTimeout小于服务端处理的时间，程序中会获得一个ReadTimeout异常。
         */
        imageAsyncManualScanRequest.setConnectTimeout(3000);
        imageAsyncManualScanRequest.setReadTimeout(10000);
        try {
            HttpResponse httpResponse = client.doAction(imageAsyncManualScanRequest);
            // 服务端接收到请求，完成处理后返回的结果。
            if (httpResponse != null && httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(org.apache.commons.codec.binary.S
tringUtils.newStringUtf8(httpResponse.getHttpContent()));
                System.out.println(JSON.toJSONString(scrResponse, true));
                int requestCode = scrResponse.getIntValue("code");
                // 每一个任务的检测结果。
                JSONArray taskResults = scrResponse.getJSONArray("data");
                if (200 == requestCode) {
                    for (Object taskResult : taskResults) {
                        // 单个任务的处理结果。
                        int taskCode = ((JSONObject) taskResult).getIntValue("code");
                        if (200 == taskCode) {
                            // 保存taskId用于轮询结果。
                            System.out.println(((JSONObject)taskResult).getString("taskId")
);
                        } else {
                            // 单个任务处理失败，原因视具体的情况详细分析。
                            System.out.println("task process fail. task response:" + JSON.t
oJSONString(taskResult));
                        }
                    }
                } else {
                    /**
                     * 表明请求整体处理失败，原因视具体的情况详细分析。
                     */
                    System.out.println("the whole scan request failed. response:" + JSON.to
JSONString(scrResponse));
                }
            }
        } catch (ServerException e) {
            e.printStackTrace();
        } catch (ClientException e) {
            e.printStackTrace();
        }
    }
}
```

查询图片人工审核任务结果

示例代码

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.green.model.v20180509.ImageAsyncManualScanResultsRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.http.MethodType;
import com.aliyuncs.http.ProtocolType;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import java.util.ArrayList;
import java.util.List;
public class ImageAsyncManualScanResults {
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile.getProfile("cn-shanghai", "您的AccessKey ID"
, "您的AccessKey Secret");
        DefaultProfile.addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com"
);
        IAcsClient client = new DefaultAcsClient(profile);
        ImageAsyncManualScanResultsRequest imageAsyncManualScanResultsRequest =
                new ImageAsyncManualScanResultsRequest();
        // 指定API返回格式。
        imageAsyncManualScanResultsRequest.setAcceptFormat(FormatType.JSON);
        // 指定请求方法。
        imageAsyncManualScanResultsRequest.setMethod(MethodType.POST);
        imageAsyncManualScanResultsRequest.setEncoding("utf-8");
        // 支持HTTP和HTTPS。
        imageAsyncManualScanResultsRequest.setProtocol(ProtocolType.HTTP);
        List<String> taskIds = new ArrayList<String>();
        // 这里添加要查询的taskId。提交任务的时候需要自行保存taskId。
        taskIds.add("图片人工审核任务ID");
        imageAsyncManualScanResultsRequest.setHttpContent(JSON.toJSONString(taskIds).getByt
es("UTF-8"), "UTF-8", FormatType.JSON);
        /**
         * 请务必设置超时时间。
         */
        imageAsyncManualScanResultsRequest.setConnectTimeout(3000);
        imageAsyncManualScanResultsRequest.setReadTimeout(6000);
        try {
            HttpResponse httpResponse = client.doAction(imageAsyncManualScanResultsRequest)
;
            if (httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttpCo
ntent(), "UTF-8"));
                System.out.println(JSON.toJSONString(scrResponse, true));
                if (200 == scrResponse.getInteger("code")) {
                    JSONArray taskResults = scrResponse.getJSONArray("data");
```

```
                    JSONArray taskResults = scrResponse.getJSONArray("data");
                    for (Object taskResult : taskResults) {
                        if (200 == ((JSONObject) taskResult).getInteger("code")) {
                            String url = ((JSONObject) taskResult).getString("url");
                            String taskId = ((JSONObject) taskResult).getString("taskId");
                            String suggestion = ((JSONObject) taskResult).getString("sugges
tion");
                            // 根据不同的suggestion结果做业务上的不同处理。例如，将违规数据删除等
。
                        } else {
                            System.out.println("task process fail:" + ((JSONObject) taskRes
ult).getInteger("code"));
                        }
                    }
                } else {
                    System.out.println("detect not success. code:" + scrResponse.getInteger
("code"));
                }
            } else {
                System.out.println("response not success. status:" + httpResponse.getStatus
());
            }
        } catch (ServerException e) {
            e.printStackTrace();
        } catch (ClientException e) {
            e.printStackTrace();
        }
    }
}
```

2.4.2. 视频人工审核

本文介绍如何使用Java SDK视频人工审核接口。

功能描述

如果您认为视频检测结果（机审）与预期不符，可以使用视频人工审核接口。关于参数的详细信息，请参见视频人工审核API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Java依赖。关于安装Java依赖的具体操作，请参见安装Java依赖。

  > **说明** 请一定按照安装Java依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

提交视频人工审核任务

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
```

```
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.green.model.v20180509.VideoAsyncManualScanRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.http.MethodType;
import com.aliyuncs.http.ProtocolType;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import java.util.Arrays;
public class VideoAsyncManualScanSample {
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile.getProfile("cn-shanghai", "您的AccessKey ID"
, "您的AccessKey Secret");
        DefaultProfile.addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com"
);
        IAcsClient client = new DefaultAcsClient(profile);
        VideoAsyncManualScanRequest videoAsyncManualScanRequest = new VideoAsyncManualScanR
equest();
        // 指定API返回格式。
        videoAsyncManualScanRequest.setAcceptFormat(FormatType.JSON);
        // 指定请求方法。
        videoAsyncManualScanRequest.setMethod(MethodType.POST);
        videoAsyncManualScanRequest.setEncoding("utf-8");
        // 支持HTTP和HTTPS。
        videoAsyncManualScanRequest.setProtocol(ProtocolType.HTTP);
        JSONObject httpBody = new JSONObject();
        JSONObject task = new JSONObject();
        task.put("dataId", "检测数据ID");
        // 设置视频链接。
        task.put("url", "待检测视频链接");
        httpBody.put("tasks", Arrays.asList(task));
        // callback、seed用于回调通知，可选参数。
        httpBody.put("callback", "回调地址");
        httpBody.put("seed", "随机字符串");
        videoAsyncManualScanRequest.setHttpContent(org.apache.commons.codec.binary.StringUt
ils.getBytesUtf8(httpBody.toJSONString()),
                "UTF-8", FormatType.JSON);
        /**
         * 请设置超时时间。服务端全链路处理超时时间为10秒，请做相应设置。
         * 如果您设置的ReadTimeout小于服务端处理的时间，程序中会获取一个ReadTimeout异常。
         */
        videoAsyncManualScanRequest.setConnectTimeout(3000);
        videoAsyncManualScanRequest.setReadTimeout(10000);
        try {
            HttpResponse httpResponse = client.doAction(videoAsyncManualScanRequest);
            // 服务端接收到请求，完成处理后返回的结果。
            if (httpResponse != null && httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(org.apache.commons.codec.binary.S
tringUtils.newStringUtf8(httpResponse.getHttpContent()));
                System.out.println(JSON.toJSONString(scrResponse, true));
```

```
                int requestCode = scrResponse.getIntValue("code");
                // 每一个任务的检测结果。
                JSONArray taskResults = scrResponse.getJSONArray("data");
                if (200 == requestCode) {
                    for (Object taskResult : taskResults) {
                        // 单个任务的处理结果。
                        int taskCode = ((JSONObject) taskResult).getIntValue("code");
                        if (200 == taskCode) {
                            // 保存taskId用于轮询结果。
                            System.out.println(((JSONObject)taskResult).getString("taskId")
);
                        } else {
                            // 单个任务处理失败，原因视具体的情况详细分析。
                            System.out.println("task process fail. task response:" + JSON.t
oJSONString(taskResult));
                        }
                    }
                } else {
                    /**
                     * 表明请求整体处理失败，原因视具体的情况详细分析。
                     */
                    System.out.println("the whole scan request failed. response:" + JSON.to
JSONString(scrResponse));
                }
            }
        } catch (ServerException e) {
            e.printStackTrace();
        } catch (ClientException e) {
            e.printStackTrace();
        }
    }
}
```

查询视频人工审核任务结果

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.green.model.v20180509.VideoAsyncManualScanResultsRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.http.MethodType;
import com.aliyuncs.http.ProtocolType;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import java.util.ArrayList;
import java.util.List;
public class VideoAsyncManualScanResults {
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile.getProfile("cn-shanghai", "您的
```

```
        IClientProfile profile = DefaultProfile.getProfile("cn-shanghai", "您的AccessKey ID"
, "您的AccessKey Secret");
        DefaultProfile.addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com"
);
        IAcsClient client = new DefaultAcsClient(profile);
        VideoAsyncManualScanResultsRequest videoAsyncManualScanResultsRequest = new VideoAs
yncManualScanResultsRequest();
        // 指定API返回格式。
        videoAsyncManualScanResultsRequest.setAcceptFormat(FormatType.JSON);
        // 指定请求方法。
        videoAsyncManualScanResultsRequest.setMethod(MethodType.POST);
        videoAsyncManualScanResultsRequest.setEncoding("utf-8");
        // 支持HTTP和HTTPS。
        videoAsyncManualScanResultsRequest.setProtocol(ProtocolType.HTTP);
        List<String> taskIds = new ArrayList<String>();
        // 这里添加要查询的taskId，提交任务的时候需要自行保存taskId。
        taskIds.add("视频人工审核任务ID");
        videoAsyncManualScanResultsRequest.setHttpContent(JSON.toJSONString(taskIds).getByt
es("UTF-8"), "UTF-8", FormatType.JSON);
        /**
         * 请务必设置超时时间。
         */
        videoAsyncManualScanResultsRequest.setConnectTimeout(3000);
        videoAsyncManualScanResultsRequest.setReadTimeout(6000);
        try {
            HttpResponse httpResponse = client.doAction(videoAsyncManualScanResultsRequest)
;
            if (httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttpCo
ntent(), "UTF-8"));
                System.out.println(JSON.toJSONString(scrResponse, true));
                if (200 == scrResponse.getInteger("code")) {
                    JSONArray taskResults = scrResponse.getJSONArray("data");
                    for (Object taskResult : taskResults) {
                        if (200 == ((JSONObject) taskResult).getInteger("code")) {
                            String url = ((JSONObject) taskResult).getString("url");
                            String taskId = ((JSONObject) taskResult).getString("taskId");
                            String suggestion = ((JSONObject) taskResult).getString("sugges
tion");
                            // 根据不同的suggestion结果做业务上的不同处理。例如，将违规数据删除等
。
                        } else {
                            System.out.println("task process fail:" + ((JSONObject) taskRes
ult).getInteger("code"));
                        }
                    }
                } else {
                    System.out.println("detect not success. code:" + scrResponse.getInteger
("code"));
                }
            } else {
                System.out.println("response not success. status:" + httpResponse.getStatus
());
            }
        } catch (ServerException e) {
```

```
            e.printStackTrace();
        } catch (ClientException e) {
            e.printStackTrace();
        }
    }
}
```

2.4.3. 文本人工审核

本文介绍如何使用Java SDK文本人工审核接口。

功能描述

如果您认为文本检测结果（机审）与预期不符，可以使用文本人工审核接口。关于参数的详细信息，请参见文本人工审核API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Java依赖。关于安装Java依赖的具体操作，请参见安装Java依赖。

  > **说明** 请一定按照安装Java依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

提交文本人工审核任务

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.green.model.v20180509.TextAsyncManualScanRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.http.MethodType;
import com.aliyuncs.http.ProtocolType;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import java.util.Arrays;
public class TextAsyncManualScanSample {
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile.getProfile("cn-shanghai", "您的AccessKey ID"
, "您的AccessKey Secret");
        DefaultProfile.addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com"
);
        IAcsClient client = new DefaultAcsClient(profile);
        TextAsyncManualScanRequest textAsyncManualScanRequest = new TextAsyncManualScanRequ
est();
```

```
        // 指定API返回格式。
        textAsyncManualScanRequest.setAcceptFormat(FormatType.JSON);
        // 指定请求方法。
        textAsyncManualScanRequest.setMethod(MethodType.POST);
        textAsyncManualScanRequest.setEncoding("utf-8");
        // 支持HTTP和HTTPS。
        textAsyncManualScanRequest.setProtocol(ProtocolType.HTTP);
        JSONObject task = new JSONObject();
        task.put("dataId", "检测数据ID");
        task.put("content", "文本内容");
        JSONObject httpBody = new JSONObject();
        httpBody.put("tasks", Arrays.asList(task));
        // callback、seed用于回调通知，可选参数。
        httpBody.put("callback", "回调地址");
        httpBody.put("seed", "随机字符串");
        textAsyncManualScanRequest.setHttpContent(org.apache.commons.codec.binary.StringUti
ls.getBytesUtf8(httpBody.toJSONString()),
                "UTF-8", FormatType.JSON);
        /**
         * 请设置超时时间。服务端全链路处理超时时间为10秒，请做相应设置。
         * 如果您设置的ReadTimeout小于服务端处理的时间，程序中会获取一个ReadTimeout异常。
         */
        textAsyncManualScanRequest.setConnectTimeout(3000);
        textAsyncManualScanRequest.setReadTimeout(10000);
        try {
            HttpResponse httpResponse = client.doAction(textAsyncManualScanRequest);
            // 服务端接收到请求，完成处理后返回的结果。
            if (httpResponse != null && httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(org.apache.commons.codec.binary.S
tringUtils.newStringUtf8(httpResponse.getHttpContent()));
                System.out.println(JSON.toJSONString(scrResponse, true));
                int requestCode = scrResponse.getIntValue("code");
                // 每一个任务的检测结果。
                JSONArray taskResults = scrResponse.getJSONArray("data");
                if (200 == requestCode) {
                    for (Object taskResult : taskResults) {
                        // 单个任务的处理结果。
                        int taskCode = ((JSONObject) taskResult).getIntValue("code");
                        if (200 == taskCode) {
                            // 保存taskId用于轮询结果。
                            System.out.println(((JSONObject)taskResult).getString("taskId")
);
                        } else {
                            // 单个任务处理失败，原因视具体的情况详细分析。
                            System.out.println("task process fail. task response:" + JSON.t
oJSONString(taskResult));
                        }
                    }
                } else {
                    /**
                     * 表明请求整体处理失败，原因视具体的情况详细分析。
                     */
                    System.out.println("the whole scan request failed. response:" + JSON.to
JSONString(scrResponse));
```

```
                }
            }
        } catch (ServerException e) {
            e.printStackTrace();
        } catch (ClientException e) {
            e.printStackTrace();
        }
    }
}
```

查询文本人工审核任务结果

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.green.model.v20180509.TextAsyncManualScanResultsRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.http.MethodType;
import com.aliyuncs.http.ProtocolType;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import java.util.ArrayList;
import java.util.List;
public class TextAsyncManualScanResults {
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile.getProfile("cn-shanghai", "您的AccessKey ID"
, "您的AccessKey Secret");
        DefaultProfile.addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com"
);
        IAcsClient client = new DefaultAcsClient(profile);
        TextAsyncManualScanResultsRequest textAsyncManualScanResultsRequest =
                new TextAsyncManualScanResultsRequest();
        // 指定API返回格式。
        textAsyncManualScanResultsRequest.setAcceptFormat(FormatType.JSON);
        // 指定请求方法。
        textAsyncManualScanResultsRequest.setMethod(MethodType.POST);
        textAsyncManualScanResultsRequest.setEncoding("utf-8");
        // 支持HTTP和HTTPS。
        textAsyncManualScanResultsRequest.setProtocol(ProtocolType.HTTP);
        List<String> taskIds = new ArrayList<String>();
        // 这里添加要查询的taskId。提交任务的时候需要自行保存taskId。
        taskIds.add("文本审核任务ID");
        textAsyncManualScanResultsRequest.setHttpContent(JSON.toJSONString(taskIds).getByte
s("UTF-8"), "UTF-8", FormatType.JSON);
        /**
         * 请务必设置超时时间。
         */
        textAsyncManualScanResultsRequest.setConnectTimeout(3000);
        textAsyncManualScanResultsRequest.setReadTimeout(6000);
```

```
        try {
            HttpResponse httpResponse = client.doAction(textAsyncManualScanResultsRequest);
            if (httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttpCo
ntent(), "UTF-8"));
                System.out.println(JSON.toJSONString(scrResponse, true));
                if (200 == scrResponse.getInteger("code")) {
                    JSONArray taskResults = scrResponse.getJSONArray("data");
                    for (Object taskResult : taskResults) {
                        if (200 == ((JSONObject) taskResult).getInteger("code")) {
                            String taskId = ((JSONObject) taskResult).getString("taskId");
                            String suggestion = ((JSONObject) taskResult).getString("sugges
tion");
                            // 根据不同的suggestion结果做业务上的不同处理。例如，将违规数据删除等
。
                        } else {
                            System.out.println("task process fail:" + ((JSONObject) taskRes
ult).getInteger("code"));
                        }
                    }
                } else {
                    System.out.println("detect not success. code:" + scrResponse.getInteger
("code"));
                }
            } else {
                System.out.println("response not success. status:" + httpResponse.getStatus
());
            }
        } catch (ServerException e) {
            e.printStackTrace();
        } catch (ClientException e) {
            e.printStackTrace();
        }
    }
}
```

2.4.4. 语音人工审核

本文介绍如何使用Java SDK语音人工审核接口。

功能描述

如果您认为语音检测结果（机审）与预期不符，可以使用语音人工审核。关于参数的详细信息，请参见语音人工审核API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Java依赖。关于安装Java依赖的具体操作，请参见安装Java依赖。

> **说明** 请一定按照安装Java依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

提交语音人工审核任务

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.green.model.v20180509.VoiceAsyncManualScanRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.http.MethodType;
import com.aliyuncs.http.ProtocolType;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import java.util.Arrays;
public class VoiceAsyncManualScanSample {
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile.getProfile("cn-shanghai", "您的AccessKey ID"
, "您的AccessKey Secret");
        DefaultProfile.addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com"
);
        IAcsClient client = new DefaultAcsClient(profile);
        VoiceAsyncManualScanRequest voiceAsyncManualScanRequest = new VoiceAsyncManualScanR
equest();
        // 指定API返回格式。
        voiceAsyncManualScanRequest.setAcceptFormat(FormatType.JSON);
        // 指定请求方法。
        voiceAsyncManualScanRequest.setMethod(MethodType.POST);
        voiceAsyncManualScanRequest.setEncoding("utf-8");
        // 支持HTTP和HTTPS。
        voiceAsyncManualScanRequest.setProtocol(ProtocolType.HTTP);
        JSONObject task = new JSONObject();
        task.put("dataId", "检测数据ID");
        // 设置链接。
        task.put("url", "待检测音频链接");
        JSONObject httpBody = new JSONObject();
        httpBody.put("tasks", Arrays.asList(task));
        // callback、seed用于回调通知，可选参数。
        httpBody.put("callback", "回调地址");
        httpBody.put("seed", "随机字符串");
        voiceAsyncManualScanRequest.setHttpContent(org.apache.commons.codec.binary.StringUt
ils.getBytesUtf8(httpBody.toJSONString()),
                "UTF-8", FormatType.JSON);
        /**
         * 请设置超时时间。服务端全链路处理超时时间为10秒，请做相应设置。
         * 如果您设置的ReadTimeout小于服务端处理的时间，程序中会获得一个ReadTimeout异常。
         */
```

```
        voiceAsyncManualScanRequest.setConnectTimeout(3000);
        voiceAsyncManualScanRequest.setReadTimeout(10000);
        try {
            HttpResponse httpResponse = client.doAction(voiceAsyncManualScanRequest);
            // 服务端接收到请求，完成处理后返回的结果。
            if (httpResponse != null && httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(org.apache.commons.codec.binary.S
tringUtils.newStringUtf8(httpResponse.getHttpContent()));
                System.out.println(JSON.toJSONString(scrResponse, true));
                int requestCode = scrResponse.getIntValue("code");
                // 每一个任务的检测结果。
                JSONArray taskResults = scrResponse.getJSONArray("data");
                if (200 == requestCode) {
                    for (Object taskResult : taskResults) {
                        // 单个任务的处理结果。
                        int taskCode = ((JSONObject) taskResult).getIntValue("code");
                        if (200 == taskCode) {
                            // 保存taskId用于轮询结果。
                            System.out.println(((JSONObject)taskResult).getString("taskId")
);
                        } else {
                            // 单个任务处理失败，原因视具体的情况详细分析。
                            System.out.println("task process fail. task response:" + JSON.t
oJSONString(taskResult));
                        }
                    }
                } else {
                    /**
                     * 表明请求整体处理失败，原因视具体的情况详细分析。
                     */
                    System.out.println("the whole scan request failed. response:" + JSON.to
JSONString(scrResponse));
                }
            }
        } catch (ServerException e) {
            e.printStackTrace();
        } catch (ClientException e) {
            e.printStackTrace();
        }
    }
}
```

查询语音人工审核任务结果

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.green.model.v20180509.ImageAsyncManualScanResultsRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
```

```
import com.aliyuncs.http.MethodType;
import com.aliyuncs.http.ProtocolType;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import java.util.ArrayList;
import java.util.List;
public class VoiceAsyncManualScanResults {
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile.getProfile("cn-shanghai", "您的AccessKey ID"
, "您的AccessKey Secret");
        DefaultProfile.addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com"
);
        IAcsClient client = new DefaultAcsClient(profile);
        ImageAsyncManualScanResultsRequest imageAsyncManualScanResultsRequest =
                new ImageAsyncManualScanResultsRequest();
        // 指定API返回格式。
        imageAsyncManualScanResultsRequest.setAcceptFormat(FormatType.JSON);
        // 指定请求方法。
        imageAsyncManualScanResultsRequest.setMethod(MethodType.POST);
        imageAsyncManualScanResultsRequest.setEncoding("utf-8");
        // 支持HTTP和HTTPS。
        imageAsyncManualScanResultsRequest.setProtocol(ProtocolType.HTTP);
        List<String> taskIds = new ArrayList<String>();
        // 这里添加要查询的taskId。提交任务的时候需要自行保存taskId。
        taskIds.add("文本审核任务ID");
        imageAsyncManualScanResultsRequest.setHttpContent(JSON.toJSONString(taskIds).getByt
es("UTF-8"), "UTF-8", FormatType.JSON);
        /**
         * 请务必设置超时时间。
         */
        imageAsyncManualScanResultsRequest.setConnectTimeout(3000);
        imageAsyncManualScanResultsRequest.setReadTimeout(6000);
        try {
            HttpResponse httpResponse = client.doAction(imageAsyncManualScanResultsRequest)
;
            if (httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttpCo
ntent(), "UTF-8"));
                System.out.println(JSON.toJSONString(scrResponse, true));
                if (200 == scrResponse.getInteger("code")) {
                    JSONArray taskResults = scrResponse.getJSONArray("data");
                    for (Object taskResult : taskResults) {
                        if (200 == ((JSONObject) taskResult).getInteger("code")) {
                            String url = ((JSONObject) taskResult).getString("url");
                            String taskId = ((JSONObject) taskResult).getString("taskId");
                            String suggestion = ((JSONObject) taskResult).getString("sugges
tion");
                            // 根据不同的suggestion结果做业务上的不同处理。例如，将违规数据删除等
。
                        } else {
                            System.out.println("task process fail:" + ((JSONObject) taskRes
ult).getInteger("code"));
                        }
                    }
```

```
                } else {
                    System.out.println("detect not success. code:" + scrResponse.getInteger
("code"));
                }
            } else {
                System.out.println("response not success. status:" + httpResponse.getStatus
());
            }
        } catch (ServerException e) {
            e.printStackTrace();
        } catch (ClientException e) {
            e.printStackTrace();
        }
    }
}
```

2.5. 图片OCR识别

本文介绍了如何使用Java SDK图片OCR接口，识别图片中的文字或卡证信息。

功能描述

通用OCR除了能够识别普通图片中的文字，还能识别结构化的卡证上的文字。关于参数的详细说明，请参见图片OCR检测API文档。

前提条件

- 安装Java依赖。关于安装Java依赖的具体操作，请参见安装Java依赖。

  > **说明** 请一定按照安装Java依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

提交图片同步检测任务

接口	描述	支持的Region
ImageSyncScanRequest	提交图片OCR同步识别任务，对图片中的文字进行识别（scene=ocr）。	• *cn-shanghai* • *cn-beijing* • *cn-shenzhen* • *ap-southeast-1*

示例代码

- 传图片URL进行检测

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
```

```
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.green.model.v20180509.ImageSyncScanRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.http.MethodType;
import com.aliyuncs.http.ProtocolType;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import java.util.*;
public class Main {
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile
            .getProfile("cn-shanghai", "yourAccessKeyId", "yourAccessKeySecret");
        DefaultProfile
            .addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        ImageSyncScanRequest imageSyncScanRequest = new ImageSyncScanRequest();
        // 指定返回格式。
        imageSyncScanRequest.setAcceptFormat(FormatType.JSON);
        // 指定请求方法。
        imageSyncScanRequest.setMethod(MethodType.POST);
        imageSyncScanRequest.setEncoding("utf-8");
        // 支持HTTP和HTTPS。
        imageSyncScanRequest.setProtocol(ProtocolType.HTTP);
        JSONObject httpBody = new JSONObject();
        /**
         * 设置要检测的场景。
         * ocr：表示OCR图文识别和OCR卡证识别。
         */
        httpBody.put("scenes", Arrays.asList("ocr"));
        /**
         * 设置待检测的图片，一张图片对应一个检测任务。
         * 多张图片同时检测时，处理时间由最后一张处理完的图片决定。
         * 通常情况下批量检测的平均响应时间比单任务检测长，一次批量提交的图片数越多，响应时间被拉长的概率越高。
         * 代码中以单张图片检测作为示例，如果需要批量检测多张图片，请自行构建多个任务。
         */
        JSONObject task = new JSONObject();
        task.put("dataId", UUID.randomUUID().toString());
        // 设置图片链接。
        task.put("url", "https://example.com/xxx.jpg");
        task.put("time", new Date());
        httpBody.put("tasks", Arrays.asList(task));
        // 需要OCR卡证识别时，设置卡证类型。
        JSONObject cardExtras = new JSONObject();
        // 身份证正面识别。
        cardExtras.put("card", "id-card-front");
        // 身份证反面。
        //cardExtras.put("card", "id-card-back");
        httpBody.put("extras", cardExtras);
        imageSyncScanRequest.setHttpContent(org.apache.commons.codec.binary.StringUtils.getBytesUtf8(httpBody.toJSONString()), "UTF-8", FormatType.JSON);
        /**
         * 请设置超时时间，服务端全链路处理超时时间为10秒，请据此做相应设置。
```

```
         * 如果您设置的ReadTimeout小于服务端处理的时间，程序中会获得一个read timeout异常。
         */
        imageSyncScanRequest.setConnectTimeout(3000);
        imageSyncScanRequest.setReadTimeout(10000);
        HttpResponse httpResponse = null;
        try {
            httpResponse = client.doAction(imageSyncScanRequest);
        } catch (ServerException e) {
            e.printStackTrace();
        } catch (ClientException e) {
            e.printStackTrace();
        } catch (Exception e){
            e.printStackTrace();
        }
        // 服务端接收到请求，并完成处理返回的结果。
        if(httpResponse != null && httpResponse.isSuccess()){
            JSONObject scrResponse = JSON.parseObject(org.apache.commons.codec.binary.Str
ingUtils.newStringUtf8(httpResponse.getHttpContent()));
            System.out.println(JSON.toJSONString(scrResponse));
            int requestCode = scrResponse.getIntValue("code");
            // 每一张图片的检测结果。
            JSONArray taskResults = scrResponse.getJSONArray("data");
            if (200 == requestCode) {
                for (Object taskResult : taskResults) {
                    // 单张图片的处理结果。
                    int taskCode = ((JSONObject)taskResult).getIntValue("code");
                    // 对应检测场景下图片的处理结果。如果是多个场景，则会有每个场景的结果。
                    JSONArray sceneResults = ((JSONObject)taskResult).getJSONArray("resul
ts");
                    if(200 == taskCode){
                        for (Object sceneResult : sceneResults) {
                            String scene = ((JSONObject)sceneResult).getString("scene");
                            String suggestion = ((JSONObject)sceneResult).getString("sugg
estion");
                            //do something
                            // 识别出来的身份证信息。
                            if("review" .equals(suggestion) && "ocr".equals(scene)){
                                JSONObject idCardInfo =  ((JSONObject) sceneResult).getJS
ONObject("idCardInfo");
                                System.out.println(idCardInfo.toJSONString());
                            }
                        }
                    }else{
                        // 单张图片处理失败，原因视具体的情况详细分析。
                        System.out.println("task process fail. task response:" + JSON.toJ
SONString(taskResult));
                     }
                }
            } else {
                /**
                 * 表明请求整体处理失败，原因视具体的情况详细分析。
                 */
                System.out.println("the whole image scan request failed. response:" + JSO
N.toJSONString(scrResponse));
```

```
            }
        }
    }
}
```

- 传本地图片文件进行检测

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.green.extension.uploader.ClientUploader;
import com.aliyuncs.green.model.v20180509.ImageSyncScanRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.http.MethodType;
import com.aliyuncs.http.ProtocolType;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import java.util.*;
public class Main {
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile
            .getProfile("cn-shanghai", "yourAccessKeyId", "yourAccessKeySecret");
        DefaultProfile
            .addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        ImageSyncScanRequest imageSyncScanRequest = new ImageSyncScanRequest();
        // 指定返回格式。
        imageSyncScanRequest.setAcceptFormat(FormatType.JSON);
        // 指定请求方法。
        imageSyncScanRequest.setMethod(MethodType.POST);
        imageSyncScanRequest.setEncoding("utf-8");
        // 支持HTTP和HTTPS。
        imageSyncScanRequest.setProtocol(ProtocolType.HTTP);
        JSONObject httpBody = new JSONObject();
        /**
         * 设置要检测的场景。
         * ocr：表示OCR图文识别和OCR卡证识别。
         */
        httpBody.put("scenes", Arrays.asList("ocr"));
        /**
         * 设置待检测的图片，一张图片对应一个检测任务。
         * 多张图片同时检测时，处理时间由最后一张处理完的图片决定。
         * 通常情况下批量检测的平均响应时间比单任务检测长，一次批量提交的图片数越多，响应时间被拉长的概率越高。
         * 代码中以单张图片检测作为示例，如果需要批量检测多张图片，请自行构建多个任务。
         * 本地图片相对于互联网图片链接，多了一个上传步骤，上传后取返回的链接进行检测。
         */
        ClientUploader clientUploader = ClientUploader.getImageClientUploader(profile, false);
        String url = null;
```

```
        try{
            url = clientUploader.uploadFile("/Users/test/Pictures/idfront-data-1536747400
633.jpg");
        }catch (Exception e){
            System.out.println("upload file to server fail.");
        }
        JSONObject task = new JSONObject();
        task.put("dataId", UUID.randomUUID().toString());
        task.put("url", url);
        task.put("time", new Date());
        httpBody.put("tasks", Arrays.asList(task));
        // 使用OCR卡证识别时，设置要识别的卡证类型。
        JSONObject cardExtras = new JSONObject();
        // 身份证正面。
        cardExtras.put("card", "id-card-front");
        // 身份证反面。
        //cardExtras.put("card", "id-card-back");
        httpBody.put("extras", cardExtras);
        imageSyncScanRequest.setHttpContent(org.apache.commons.codec.binary.StringUtils.g
etBytesUtf8(httpBody.toJSONString()), "UTF-8", FormatType.JSON);
        /**
         * 请设置超时时间。服务端全链路处理超时时间为10秒，请据此做相应设置。
         * 如果您设置的ReadTimeout小于服务端处理的时间，程序中会获得一个read timeout异常。
         */
        imageSyncScanRequest.setConnectTimeout(3000);
        imageSyncScanRequest.setReadTimeout(10000);
        HttpResponse httpResponse = null;
        try {
            httpResponse = client.doAction(imageSyncScanRequest);
        } catch (ServerException e) {
            e.printStackTrace();
        } catch (ClientException e) {
            e.printStackTrace();
        } catch (Exception e){
            e.printStackTrace();
        }
        // 服务端接收到请求，并完成处理返回的结果。
        if(httpResponse != null && httpResponse.isSuccess()){
            JSONObject scrResponse = JSON.parseObject(org.apache.commons.codec.binary.Str
ingUtils.newStringUtf8(httpResponse.getHttpContent()));
            System.out.println(JSON.toJSONString(scrResponse));
            int requestCode = scrResponse.getIntValue("code");
            // 每一张图片的检测结果。
            JSONArray taskResults = scrResponse.getJSONArray("data");
            if (200 == requestCode) {
                for (Object taskResult : taskResults) {
                    // 单张图片的处理结果。
                    int taskCode = ((JSONObject)taskResult).getIntValue("code");
                    // 对应检测场景下，图片的处理结果。如果是多个场景，则会有每个场景的结果。
                    JSONArray sceneResults = ((JSONObject)taskResult).getJSONArray("resul
ts");
                    if(200 == taskCode){
                        for (Object sceneResult : sceneResults) {
                            String scene = ((JSONObject)sceneResult).getString("scene");
```

```
                            String suggestion = ((JSONObject)sceneResult).getString("sugg
estion");
                            //do something
                            // 识别出来的卡证信息。
                            if("review" .equals(suggestion) && "ocr".equals(scene)){
                                JSONObject idCardInfo =  ((JSONObject) sceneResult).getJS
ONObject("idCardInfo");
                                System.out.println(idCardInfo.toJSONString());
                            }
                        }
                    }else{
                        // 单张图片处理失败，原因视具体的情况详细分析。
                        System.out.println("task process fail. task response:" + JSON.toJ
SONString(taskResult));
                    }
                }
            } else {
                /**
                 * 表明请求整体处理失败，原因视具体的情况详细分析。
                 */
                System.out.println("the whole image scan request failed. response:" + JSO
N.toJSONString(scrResponse));
            }
        }
    }
}
```

- 传图片二进制字节组数据进行检测

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.green.extension.uploader.ClientUploader;
import com.aliyuncs.green.model.v20180509.ImageSyncScanRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.http.MethodType;
import com.aliyuncs.http.ProtocolType;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import org.apache.commons.io.FileUtils;
import java.io.File;
import java.util.*;
public class Main {
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile
            .getProfile("cn-shanghai", "yourAccessKeyId", "yourAccessKeySecret");
        DefaultProfile
            .addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        ImageSyncScanRequest imageSyncScanRequest = new ImageSyncScanRequest();
```

```
        // 指定返回格式。
        imageSyncScanRequest.setAcceptFormat(FormatType.JSON);
        // 指定请求方法。
        imageSyncScanRequest.setMethod(MethodType.POST);
        imageSyncScanRequest.setEncoding("utf-8");
        // 支持HTTP和HTTPS。
        imageSyncScanRequest.setProtocol(ProtocolType.HTTP);
        JSONObject httpBody = new JSONObject();
        /**
         * 设置要检测的场景。
         * ocr：表示OCR图文识别和OCR卡证识别。
         */
        httpBody.put("scenes", Arrays.asList("ocr"));
        /**
         * 设置待检测的图片，一张图片对应一个检测任务。
         * 多张图片同时检测时，处理时间由最后一张处理完的图片决定。
         * 通常情况下批量检测的平均响应时间比单任务检测长，一次批量提交的图片数越多，响应时间被拉长的概率越高。
         * 代码中以单张图片检测作为示例，如果需要批量检测多张图片，请自行构建多个任务。
         * 图片二进制数据检测相对于互联网图片链接，多了一个上传步骤，上传后取返回的链接进行检测。
         */
        ClientUploader clientUploader = ClientUploader.getImageClientUploader(profile, false);
        byte[] imageBytes = null;
        String url = null;
        try{
            // 读取本地文件作为二进制数据，做为输入示例。实际使用中请直接替换成您的图片二进制数据。
            imageBytes = FileUtils.readFileToByteArray(new File("/Users/01fb4ab6420b5f34623e13b82b51ef87.jpg"));
            // 上传到服务端。
            url = clientUploader.uploadBytes(imageBytes);
        }catch (Exception e){
            System.out.println(("upload file to server fail.");
        }
        JSONObject task = new JSONObject();
        task.put("dataId", UUID.randomUUID().toString());
        task.put("url", url);
        task.put("time", new Date());
        httpBody.put("tasks", Arrays.asList(task));
        // 使用OCR卡证识别时，设置要识别的卡证类型。
        JSONObject cardExtras = new JSONObject();
        // 身份证正面。
        cardExtras.put("card", "id-card-front");
        // 身份证反面。
        //cardExtras.put("card", "id-card-back");
        httpBody.put("extras", cardExtras);
        imageSyncScanRequest.setHttpContent(org.apache.commons.codec.binary.StringUtils.getBytesUtf8(httpBody.toJSONString()), "UTF-8", FormatType.JSON);
        /**
         * 请设置超时时间。服务端全链路处理超时时间为10秒，请据此做相应设置。
         * 如果您设置的ReadTimeout小于服务端处理的时间，程序中会获得一个read timeout异常。
         */
        imageSyncScanRequest.setConnectTimeout(3000);
        imageSyncScanRequest.setReadTimeout(10000);
```

```
        HttpResponse httpResponse = null;
        try {
            httpResponse = client.doAction(imageSyncScanRequest);
        } catch (ServerException e) {
            e.printStackTrace();
        } catch (ClientException e) {
            e.printStackTrace();
        } catch (Exception e){
            e.printStackTrace();
        }
        // 服务端接收到请求，并完成处理返回的结果。
        if(httpResponse != null && httpResponse.isSuccess()){
            JSONObject scrResponse = JSON.parseObject(org.apache.commons.codec.binary.Str
ingUtils.newStringUtf8(httpResponse.getHttpContent()));
            System.out.println(JSON.toJSONString(scrResponse));
            int requestCode = scrResponse.getIntValue("code");
            // 每一张图片的检测结果。
            JSONArray taskResults = scrResponse.getJSONArray("data");
            if (200 == requestCode) {
                for (Object taskResult : taskResults) {
                    // 单张图片的处理结果。
                    int taskCode = ((JSONObject)taskResult).getIntValue("code");
                    // 对应检测场景下，图片的处理结果。如果是多个场景，则会有每个场景的结果。
                    JSONArray sceneResults = ((JSONObject)taskResult).getJSONArray("resul
ts");
                    if(200 == taskCode){
                        for (Object sceneResult : sceneResults) {
                            String scene = ((JSONObject)sceneResult).getString("scene");
                            String suggestion = ((JSONObject)sceneResult).getString("sugg
estion");
                            //do something
                            // 识别出来的卡证信息。
                            if("review" .equals(suggestion) && "ocr".equals(scene)){
                                JSONObject idCardInfo =  ((JSONObject) sceneResult).getJS
ONObject("idCardInfo");
                                System.out.println(idCardInfo.toJSONString());
                            }
                        }
                    }else{
                        // 单张图片处理失败，原因视具体的情况详细分析。
                        System.out.println("task process fail. task response:" + JSON.toJ
SONString(taskResult));
                    }
                }
            } else {
                /**
                 * 表明请求整体处理失败，原因视具体的情况详细分析。
                 */
                System.out.println("the whole image scan request failed. response:" + JSO
N.toJSONString(scrResponse));
            }
        }
    }
}
```

2.6. 人脸识别

2.6.1. 人脸比对

本文介绍了如何使用Java SDK人脸比对接口，对指定的人脸图片进行属性检测。人脸属性检测仅支持以图片作为检测媒介。

功能描述

人脸比对支持同步检测和异步检测两种方式。

- 同步检测实时返回检测结果。关于参数的详细信息，请参见同步检测。
- 异步检测需要您轮询结果或者通过callback回调通知获取检测结果。关于参数的详细信息，请参见异步检测。

前提条件

- 安装Java依赖。关于安装Java依赖的具体操作，请参见安装Java依赖。

  > **说明** 请一定按照安装Java依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

传入URL示例代码

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.green.extension.uploader.ClientUploader;
import com.aliyuncs.green.model.v20180509.ImageSyncScanRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.http.MethodType;
import com.aliyuncs.http.ProtocolType;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import java.util.*;
public class ImageSyncScanSample {
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile
            .getProfile("cn-shanghai", "您的AccessKey ID", "您的AccessKey Secret");
        DefaultProfile
            .addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        // 指定API返回格式、指定请求方法。
        ImageSyncScanRequest imageSyncScanRequest = new ImageSyncScanRequest();
        imageSyncScanRequest.setAcceptFormat(FormatType.JSON);
        imageSyncScanRequest.setMethod(MethodType.POST);
        imageSyncScanRequest.setEncoding("utf-8");
        imageSyncScanRequest.setProtocol(ProtocolType.HTTP);
```

```
        JSONObject httpBody = new JSONObject();
        httpBody.put("scenes", Arrays.asList("sface-1"));
        JSONObject task = new JSONObject();
        // 设置待比对人脸图片1链接。
        task.put("url","http://example.com/xxx.jpg");
        JSONObject extras = new JSONObject();
        // 设置待比对人脸图片2链接。
        extras.put("faceUrl", "http://example.com/yyyy.jpg");
        task.put("extras", extras);
        httpBody.put("tasks", Arrays.asList(task));
        imageSyncScanRequest.setHttpContent(org.apache.commons.codec.binary.StringUtils.get
BytesUtf8(httpBody.toJSONString()),
            "UTF-8", FormatType.JSON);
        /**
         * 请设置超时时间，服务端全链路处理超时时间为10秒，请做相应设置。
         * 如果您设置的ReadTimeout小于服务端处理的时间，程序中会获得一个ReadTimeout异常。
         */
        imageSyncScanRequest.setConnectTimeout(3000);
        imageSyncScanRequest.setReadTimeout(10000);
        HttpResponse httpResponse = null;
        try {
            httpResponse = client.doAction(imageSyncScanRequest);
        } catch (Exception e) {
            e.printStackTrace();
        }
        // 服务端接收到请求，并完成处理返回的结果。
        if (httpResponse != null && httpResponse.isSuccess()) {
            JSONObject scrResponse = JSON.parseObject(org.apache.commons.codec.binary.Strin
gUtils.newStringUtf8(httpResponse.getHttpContent()));
            System.out.println(JSON.toJSONString(scrResponse, true));
            int requestCode = scrResponse.getIntValue("code");
            // 每一张图片的检测结果。
            JSONArray taskResults = scrResponse.getJSONArray("data");
            if (200 == requestCode) {
                for (Object taskResult : taskResults) {
                    // 单张图片的处理结果。
                    int taskCode = ((JSONObject) taskResult).getIntValue("code");
                    // 图片要检测的场景的处理结果，如果是多个场景，则会有每个场景的结果。
                    JSONArray sceneResults = ((JSONObject) taskResult).getJSONArray("result
s");
                    if (200 == taskCode) {
                        for (Object sceneResult : sceneResults) {
                            String scene = ((JSONObject) sceneResult).getString("scene");
                            String suggestion = ((JSONObject) sceneResult).getString("sugge
stion");
                            // 根据scene和suggetion做相关处理。
                            System.out.println("suggestion = [" + suggestion + "]");
                        }
                    } else {
                        // 单张图片处理失败，原因视具体的情况详细分析。
                        System.out.println("task process fail. task response:" + JSON.toJSO
NString(taskResult));
                    }
                }
```

```
            } else {
                /**
                 * 表明请求整体处理失败，原因视具体的情况详细分析。
                 */
                System.out.println("the whole image scan request failed. response:" + JSON.
toJSONString(scrResponse));
            }
        }
    }
}
```

传入本地文件示例代码

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.green.extension.uploader.ClientUploader;
import com.aliyuncs.green.model.v20180509.ImageSyncScanRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.http.MethodType;
import com.aliyuncs.http.ProtocolType;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import java.util.*;
public class ImageSyncScanSample {
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile
            .getProfile("cn-shanghai", "您的AccessKey ID", "您的AccessKey Secret");
        DefaultProfile
            .addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        // 指定API返回格式、指定请求方法。
        ImageSyncScanRequest imageSyncScanRequest = new ImageSyncScanRequest();
        imageSyncScanRequest.setAcceptFormat(FormatType.JSON);
        imageSyncScanRequest.setMethod(MethodType.POST);
        imageSyncScanRequest.setEncoding("utf-8");
        imageSyncScanRequest.setProtocol(ProtocolType.HTTP);
        JSONObject httpBody = new JSONObject();
        httpBody.put("scenes", Arrays.asList("sface-1"));
        ClientUploader clientUploader = ClientUploader.getImageClientUploader(profile, fals
e);
        JSONObject task = new JSONObject();
        // 上传待比对人脸图片1。
        String url1 = null;
        try{
            url1 = clientUploader.uploadFile("d:/image1.jpg");
        }catch (Exception e){
            e.printStackTrace();
        }
        task.put("url", url1);
        JSONObject extras = new JSONObject();
```

```
        JSONObject extras = new JSONObject();
        // 上传待比对人脸图片2。
        String url2 = null;
        try{
            url2 = clientUploader.uploadFile("d:/image2.jpg");
        }catch (Exception e){
            e.printStackTrace();
        }
        extras.put("faceUrl", url2);
        task.put("extras", extras);
        httpBody.put("tasks", Arrays.asList(task));
        imageSyncScanRequest.setHttpContent(org.apache.commons.codec.binary.StringUtils.get
BytesUtf8(httpBody.toJSONString()),
            "UTF-8", FormatType.JSON);
        /**
         * 请设置超时时间，服务端全链路处理超时时间为10秒，请做相应设置。
         * 如果您设置的ReadTimeout小于服务端处理的时间，程序中会获得一个ReadTimeout异常。
         */
        imageSyncScanRequest.setConnectTimeout(3000);
        imageSyncScanRequest.setReadTimeout(10000);
        HttpResponse httpResponse = null;
        try {
            httpResponse = client.doAction(imageSyncScanRequest);
        } catch (Exception e) {
            e.printStackTrace();
        }
        // 服务端接收到请求，并完成处理返回的结果。
        if (httpResponse != null && httpResponse.isSuccess()) {
            JSONObject scrResponse = JSON.parseObject(org.apache.commons.codec.binary.Strin
gUtils.newStringUtf8(httpResponse.getHttpContent()));
            System.out.println(JSON.toJSONString(scrResponse, true));
            int requestCode = scrResponse.getIntValue("code");
            // 每一张图片的检测结果。
            JSONArray taskResults = scrResponse.getJSONArray("data");
            if (200 == requestCode) {
                for (Object taskResult : taskResults) {
                    // 单张图片的处理结果。
                    int taskCode = ((JSONObject) taskResult).getIntValue("code");
                    // 图片要检测的场景的处理结果，如果是多个场景，则会有每个场景的结果。
                    JSONArray sceneResults = ((JSONObject) taskResult).getJSONArray("result
s");
                    if (200 == taskCode) {
                        for (Object sceneResult : sceneResults) {
                            String scene = ((JSONObject) sceneResult).getString("scene");
                            String suggestion = ((JSONObject) sceneResult).getString("sugge
stion");
                            // 根据scene和suggetion做相关处理。
                            System.out.println("suggestion = [" + suggestion + "]");
                        }
                    } else {
                        // 单张图片处理失败，原因视具体的情况详细分析。
                        System.out.println("task process fail. task response:" + JSON.toJSO
NString(taskResult));
                    }
                }
```

```
            } else {
                /**
                 * 表明请求整体处理失败，原因视具体的情况详细分析。
                 */
                System.out.println("the whole image scan request failed. response:" + JSON.
toJSONString(scrResponse));
            }
        }
    }
}
```

2.6.2. 人脸属性检测

本文介绍了如何使用Java SDK人脸属性检测接口，对指定的人脸图片进行属性检测。人脸属性检测仅支持以图片作为检测媒介。

功能描述

人脸属性检测能够识别图片中的人脸属性信息，包括人脸模糊度、人脸角度、人脸位置、微笑程度、是否戴眼镜、是否戴口罩、是否戴帽子、是否有胡子、是否有刘海、头发类型等。关于参数的详细说明，请参见人脸属性检测API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Java依赖。关于安装Java依赖的具体操作，请参见安装Java依赖。

  > **说明** 请一定按照安装Java依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

传入URL示例代码

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.green.model.v20180509.DetectFaceRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.http.MethodType;
import com.aliyuncs.http.ProtocolType;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
/**
 * 人脸属性检测代码示例。
 */
public class DetectFaceSample {
```

```
    public static void main(String[] args) throws Exception {
        // 请替换成您的AccessKey ID、AccessKey Secret。
        IClientProfile profile = DefaultProfile
                .getProfile("cn-shanghai", "您的AccessKey ID", "您的AccessKey Secret");
        DefaultProfile
                .addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        DetectFaceRequest detectFaceRequest = new DetectFaceRequest();
        detectFaceRequest.setAcceptFormat(FormatType.JSON); // 指定API返回格式。
        detectFaceRequest.setMethod(MethodType.POST); // 指定请求方法。
        detectFaceRequest.setEncoding("utf-8");
        detectFaceRequest.setHttpContentType(FormatType.JSON);
        JSONObject data = new JSONObject();
        // 设置待检测的人脸图片。
        data.put("url", "http://www.aliyundoc.com/xx");
        detectFaceRequest.setHttpContent(data.toJSONString().getBytes("UTF-8"), "UTF-8", Fo
rmatType.JSON);
        /**
         * 请设置超时时间，服务端全链路处理超时时间为10秒，请做相应设置。
         * 如果您设置的ReadTimeout小于服务端处理的时间，程序中会获得一个ReadTimeout异常。
         */
        detectFaceRequest.setConnectTimeout(3000);
        detectFaceRequest.setReadTimeout(10000);
        HttpResponse httpResponse;
        try {
            httpResponse = client.doAction(detectFaceRequest);
        } catch (Exception e) {
            e.printStackTrace();
            // 异常处理。
            return;
        }
        // 判断HTTP请求是否成功。
        if (httpResponse.isSuccess()) {
            JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttpConten
t(), "UTF-8"));
            if (200 == scrResponse.getInteger("code")) {
                JSONObject result = scrResponse.getJSONObject("data");
                if (200 == result.getInteger("code")) {
                    /**
                     *   结果数据。
                     */
                    JSONArray faces = result.getJSONArray("faces");
                    // 检测到的人脸个数。
                    System.out.println("faceCount:" + faces.size());
                    // 每个人脸的结果详情。
                    for (int i = 0; i < faces.size(); i++) {
                        System.out.println("face" + i + ": " + faces.getJSONObject(i).toJSO
NString());
                    }
                } else {
                    System.out.println("result fail: " + result.toJSONString());
                }
            } else {
                /**
```

```
                 * 请求整体处理失败，原因视具体的情况详细分析。
                 */
                System.out.println("request fail:" + scrResponse.toJSONString());
            }
        } else {
            /**
             * HTTP请求失败，错误详情。
             */
            System.out.println("response fail!" +
                    "\ncode:" + httpResponse.getStatus() +
                    "\nmsg:" + new String(httpResponse.getHttpContent(), "UTF-8"));
        }
    }
}
```

传入本地文件示例代码

> **说明** 如果您需要检测本地文件或者二进制文件，请下载并在项目工程中引入Extension.Uploader工具类。

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.green.extension.uploader.ClientUploader;
import com.aliyuncs.green.model.v20180509.DetectFaceRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.http.MethodType;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
/**
 * 人脸属性检测：本地图片文件检测样例。
 */
public class DetectFaceFileSample {
    public static void main(String[] args) throws Exception {
        //请替换成您的AccessKey ID、AccessKey Secret。
        IClientProfile profile = DefaultProfile
                .getProfile("cn-shanghai", "您的AccessKey ID", "您的AccessKey Secret");
        DefaultProfile
                .addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        DetectFaceRequest detectFaceRequest = new DetectFaceRequest();
        detectFaceRequest.setAcceptFormat(FormatType.JSON); // 指定API返回格式。
        detectFaceRequest.setMethod(MethodType.POST); // 指定请求方法。
        detectFaceRequest.setEncoding("utf-8");
        detectFaceRequest.setHttpContentType(FormatType.JSON);
        ClientUploader clientUploader = ClientUploader.getImageClientUploader(profile, fals
e);
        String url1 = null;
        try{
            //上传待检测人脸图片
```

```
            //上传待检测人脸图片。
            url1 = clientUploader.uploadFile("d:/image1.jpg");
        }catch (Exception e){
            e.printStackTrace();
        }
        JSONObject data = new JSONObject();
        // 设置待检测的人脸图片。
        data.put("url", url1);
        detectFaceRequest.setHttpContent(data.toJSONString().getBytes("UTF-8"), "UTF-8", Fo
rmatType.JSON);
        /**
         * 请设置超时时间，服务端全链路处理超时时间为10秒，请做相应设置。
         * 如果您设置的ReadTimeout小于服务端处理的时间，程序中会获得一个ReadTimeout异常。
         */
        detectFaceRequest.setConnectTimeout(3000);
        detectFaceRequest.setReadTimeout(10000);
        HttpResponse httpResponse;
        try {
            httpResponse = client.doAction(detectFaceRequest);
        } catch (Exception e) {
            e.printStackTrace();
            // 异常处理。
            return;
        }
        // 判断HTTP请求是否成功。
        if (httpResponse.isSuccess()) {
            JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttpConten
t(), "UTF-8"));
            if (200 == scrResponse.getInteger("code")) {
                JSONObject result = scrResponse.getJSONObject("data");
                if (200 == result.getInteger("code")) {
                    /**
                     *   结果数据。
                     */
                    JSONArray faces = result.getJSONArray("faces");
                    // 检测到的人脸个数。
                    System.out.println("faceCount:" + faces.size());
                    // 每个人脸的结果详情。
                    for (int i = 0; i < faces.size(); i++) {
                        System.out.println("face" + i + ": " + faces.getJSONObject(i).toJSO
NString());
                    }
                } else {
                    System.out.println("result fail: " + result.toJSONString());
                }
            } else {
                /**
                 * 请求整体处理失败，原因视具体的情况详细分析。
                 */
                System.out.println("request fail:" + scrResponse.toJSONString());
            }
        } else {
            /**
             * HTTP 请求失败，错误详情。
             */
```

```
            System.out.println("response fail!" +
                    "\ncode:" + httpResponse.getStatus() +
                    "\nmsg:" + new String(httpResponse.getHttpContent(), "UTF-8"));
        }
    }
}
```

2.6.3. 自定义人脸检索

2.6.3.1. 自定义人脸检索

本文介绍了如何使用Java SDK自定义人脸检索接口，对指定的人脸图片进行自定义人脸检索。

功能描述

自定义人脸检索根据您输入的待识别人脸图片（face），在指定人脸库（group）中查找并返回最相似的Top 5个体（person），返回的Top 5个体按照相似度从大到小排序。

检索人脸时，必须指定要检索的分组信息（最多指定一个分组）。关于参数的详细说明，请参见自定义人脸检索API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Java依赖。关于安装Java依赖的具体操作，请参见安装Java依赖。

  > **说明** 请一定按照安装Java依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

人脸图片检索（URL）代码示例

```
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Date;
import java.util.HashMap;
import java.util.LinkedHashMap;
import java.util.List;
import java.util.Map;
import java.util.UUID;
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.green.model.v20180509.ImageSyncScanRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.profile.DefaultProfile;
```

```
import com.aliyuncs.profile.IClientProfile;
/**
 * 人脸检索。
 * @author
 * @date 2018/06/25
 */
public class FaceScanRequestSample {
    public static void main(String[] args) throws Exception {
        // 请替换成您自己的AccessKey ID、AccessKey Secret。
        IClientProfile profile = DefaultProfile.getProfile("cn-shanghai", "您的AccessKey ID"
, "您的AccessKey Secret");
        DefaultProfile.addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com"
);
        IAcsClient client = new DefaultAcsClient(profile);
        ImageSyncScanRequest imageSyncScanRequest = new ImageSyncScanRequest();
        imageSyncScanRequest.setAcceptFormat(FormatType.JSON); // 指定API返回格式。
        imageSyncScanRequest.setMethod(com.aliyuncs.http.MethodType.POST); // 指定请求方法。
        imageSyncScanRequest.setEncoding("utf-8");
        imageSyncScanRequest.setRegionId("cn-shanghai");
        List<Map<String, Object>> tasks = new ArrayList<Map<String, Object>>();
        Map<String, Object> task = new LinkedHashMap<String, Object>();
        task.put("dataId", UUID.randomUUID().toString());
        task.put("url", "https://example.com/tfs/xxx-550-407.jpg");
        task.put("time", new Date());
        Map<String, String> extras = new HashMap<String, String>();
        extras.put("groupId", "java_sdk_test_group");
        task.put("extras", extras);
        tasks.add(task);
        JSONObject data = new JSONObject();
        /**
         * sface-n: 人脸1-N
         */
        data.put("scenes", Arrays.asList("sface-n"));
        data.put("tasks", tasks);
        imageSyncScanRequest.setHttpContent(data.toJSONString().getBytes("UTF-8"), "UTF-8",
FormatType.JSON);
        /**
         * 请务必设置超时时间。
         */
        imageSyncScanRequest.setConnectTimeout(3000);
        imageSyncScanRequest.setReadTimeout(6000);
        try {
            HttpResponse httpResponse = client.doAction(imageSyncScanRequest);
            if (httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttpCo
ntent(), "UTF-8"));
                System.out.println(JSON.toJSONString(scrResponse, true));
                if (200 == scrResponse.getInteger("code")) {
                    JSONArray taskResults = scrResponse.getJSONArray("data");
                    for (Object taskResult : taskResults) {
                        if(200 == ((JSONObject)taskResult).getInteger("code")){
                            JSONArray sceneResults = ((JSONObject)taskResult).getJSONArray(
"results");
                            for (Object sceneResult : sceneResults) {
```

```
                                String scene = ((JSONObject)sceneResult).getString("scene")
;
                                String suggestion = ((JSONObject)sceneResult).getString("su
ggestion");
                                // 根据scene和suggetion做相关的处理。
                                System.out.println("args = [" + scene + "]");
                                System.out.println("args = [" + suggestion + "]");
                            }
                        }else{
                            System.out.println("task process fail:" + ((JSONObject)taskResu
lt).getInteger("code"));
                        }
                    }
                } else {
                    System.out.println("detect not success. code:" + scrResponse.getInteger
("code"));
                }
            } else {
                System.out.println("response not success. status:" + httpResponse.getStatus
());
            }
        } catch (ServerException e) {
            e.printStackTrace();
        } catch (ClientException e) {
            e.printStackTrace();
        } catch (Exception e){
            e.printStackTrace();
        }
    }
}
```

人脸图片检索（本地文件）代码示例

```
import java.util.ArrayList;
import java.util.Arrays;
import java.util.Date;
import java.util.HashMap;
import java.util.LinkedHashMap;
import java.util.List;
import java.util.Map;
import java.util.UUID;
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.green.model.v20180509.ImageSyncScanRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import com.aliyuncs.green.extension.uploader.ClientUploader;
```

```
/**
 * 人脸检索(本地文件)。
 */
public class FaceScanLocalSample {
    public static void main(String[] args) throws Exception {
        // 请替换成您的AccessKey ID、AccessKey Secret。
        IClientProfile profile = DefaultProfile.getProfile("cn-shanghai", "yourAccessKeyId"
, "yourAccessKeySecret");
        DefaultProfile.addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com"
);
        IAcsClient client = new DefaultAcsClient(profile);
        ImageSyncScanRequest imageSyncScanRequest = new ImageSyncScanRequest();
        imageSyncScanRequest.setAcceptFormat(FormatType.JSON); // 指定API返回格式。
        imageSyncScanRequest.setMethod(com.aliyuncs.http.MethodType.POST); // 指定请求方法。
        imageSyncScanRequest.setEncoding("utf-8");
        imageSyncScanRequest.setRegionId("cn-shanghai");
        ClientUploader uploader = ClientUploader.getImageClientUploader(profile, false);
        String url = null;
        try {
            url = uploader.uploadFile("d:/image.jpg");
        } catch (Exception e) {
            e.printStackTrace();
        }
        List<Map<String, Object>> tasks = new ArrayList<Map<String, Object>>();
        Map<String, Object> task = new LinkedHashMap<String, Object>();
        task.put("dataId", UUID.randomUUID().toString());
        task.put("url", url);
        task.put("time", new Date());
        Map<String, String> extras = new HashMap<String, String>();
        extras.put("groupId", "java_sdk_test_group");
        task.put("extras", extras);
        tasks.add(task);
        JSONObject data = new JSONObject();
        /**
         * sface-n: 人脸1-N。
         */
        data.put("scenes", Arrays.asList("sface-n"));
        data.put("tasks", tasks);
        imageSyncScanRequest.setHttpContent(data.toJSONString().getBytes("UTF-8"), "UTF-8",
FormatType.JSON);
        /**
         * 请务必设置超时时间。
         */
        imageSyncScanRequest.setConnectTimeout(3000);
        imageSyncScanRequest.setReadTimeout(6000);
        try {
            HttpResponse httpResponse = client.doAction(imageSyncScanRequest);
            if (httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttpCo
ntent(), "UTF-8"));
                System.out.println(JSON.toJSONString(scrResponse, true));
                if (200 == scrResponse.getInteger("code")) {
                    JSONArray taskResults = scrResponse.getJSONArray("data");
                    for (Object taskResult : taskResults) {
```

```
                        if (200 == ((JSONObject) taskResult).getInteger("code")) {
                            JSONArray sceneResults = ((JSONObject) taskResult).getJSONArray
("results");
                            for (Object sceneResult : sceneResults) {
                                String scene = ((JSONObject) sceneResult).getString("scene"
);
                                String suggestion = ((JSONObject) sceneResult).getString("s
uggestion");
                                // 根据scene和suggetion做相关的处理。
                                System.out.println("args = [" + scene + "]");
                                System.out.println("args = [" + suggestion + "]");
                            }
                        } else {
                            System.out.println("task process fail:" + ((JSONObject) taskRes
ult).getInteger("code"));
                        }
                    }
                } else {
                    System.out.println("detect not success. code:" + scrResponse.getInteger
("code"));
                }
            } else {
                System.out.println("response not success. status:" + httpResponse.getStatus
());
            }
        } catch (ServerException e) {
            e.printStackTrace();
        } catch (ClientException e) {
            e.printStackTrace();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

2.6.3.2. 新建个体

本文介绍了如何使用Java SDK新建个体信息。

功能描述

新建个体时，必须指定个体要加入的分组。关于参数的详细说明，请参见新建个体API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Java依赖。关于安装Java依赖的具体操作，请参见安装Java依赖。

  > **说明** 请一定按照安装Java依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

新建个体代码示例

```
import java.util.Arrays;
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.green.model.v20180509.AddPersonRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
/**
 * 增加个体。
 * @author
 * @date 2018/06/25
 */
public class FaceAddPersonRequestSample {
    public static void main(String[] args) throws Exception {
        //请替换成您的AaccessKey ID、AccessKey Secret。
        IClientProfile profile = DefaultProfile.getProfile("cn-shanghai", "您的AaccessKey ID
", "您的AccessKey Secret");
        DefaultProfile.addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com"
);
        IAcsClient client = new DefaultAcsClient(profile);
        AddPersonRequest addPersonRequest = new AddPersonRequest();
        addPersonRequest.setAcceptFormat(FormatType.JSON); // 指定API返回格式。
        addPersonRequest.setMethod(com.aliyuncs.http.MethodType.POST); // 指定请求方法。
        addPersonRequest.setEncoding("utf-8");
        JSONObject data = new JSONObject();
        /**
         * personId: 用户自定义个体ID，必填。
         * groupIds: 用户自定义组ID列表，必填。
         * name: 用户名称，非必填。
         * note: 备注信息，非必填。
         */
        data.put("personId", "personId_test_3");
        data.put("groupIds", Arrays.asList("java_sdk_test_group"));
        data.put("name", "测试");
        data.put("note", "备注信息");
        addPersonRequest.setHttpContent(data.toJSONString().getBytes("UTF-8"), "UTF-8", For
matType.JSON);
        /**
         * 请务必设置超时时间。
         */
        addPersonRequest.setConnectTimeout(3000);
        addPersonRequest.setReadTimeout(6000);
        try {
            HttpResponse httpResponse = client.doAction(addPersonRequest);
            if (httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttpCo
ntent(), "UTF-8"));
```

```
                System.out.println(JSON.toJSONString(scrResponse, true));
                if (200 == scrResponse.getInteger("code")) {
                    JSONObject resultObject = scrResponse.getJSONObject("data");
                    if (200 == resultObject.getInteger("code")) {
                        System.out.println(resultObject.getString("personId"));
                    } else {
                        System.out.println("task process fail:" + resultObject.getInteger("
code"));
                    }
                } else {
                    System.out.println("detect not success. code:" + scrResponse.getInteger
("code"));
                }
            } else {
                System.out.println("response not success. status:" + httpResponse.getStatus
());
            }
        } catch (ServerException e) {
            e.printStackTrace();
        } catch (ClientException e) {
            e.printStackTrace();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

2.6.3.3. 设置个体

本文介绍了如何使用Java SDK设置个体信息。

功能描述

设置个体信息用于给个体添加备注信息，属于可选步骤。如果您设置了个体信息，则在自定义检索的返回结果中会包含该信息。关于参数的详细说明，请参见设置个体API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Java依赖。关于安装Java依赖的具体操作，请参见安装Java依赖。

  > **说明** 请一定按照安装Java依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

设置个体代码示例

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
```

```
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.green.model.v20180509.SetPersonRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
/**
 * 设置个体信息。
 * @author
 * @date 2018/06/25
 */
public class FaceSetPersonRequestSample {
    public static void main(String[] args) throws Exception {
        //请替换成您的AccessKey ID、AccessKey Secret。
        IClientProfile profile = DefaultProfile.getProfile("cn-shanghai", "您的AccessKey ID"
, "您的AccessKey Secret");
        DefaultProfile.addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com"
);
        IAcsClient client = new DefaultAcsClient(profile);
        SetPersonRequest setPersonRequest = new SetPersonRequest();
        setPersonRequest.setAcceptFormat(FormatType.JSON); // 指定API返回格式。
        setPersonRequest.setMethod(com.aliyuncs.http.MethodType.POST); // 指定请求方法。
        setPersonRequest.setEncoding("utf-8");
        JSONObject data = new JSONObject();
        /**
         * personId: 用户自定义个体ID，必填。
         * name: 用户名称，非必填。
         * note: 备注信息，非必填。
         */
        data.put("personId", "personId_test_3");
        data.put("name", "测试");
        data.put("note", "备注信息");
        setPersonRequest.setHttpContent(data.toJSONString().getBytes("UTF-8"), "UTF-8", For
matType.JSON);
        /**
         * 请务必设置超时时间。
         */
        setPersonRequest.setConnectTimeout(3000);
        setPersonRequest.setReadTimeout(6000);
        try {
            HttpResponse httpResponse = client.doAction(setPersonRequest);
            if (httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttpCo
ntent(), "UTF-8"));
                System.out.println(JSON.toJSONString(scrResponse, true));
                if (200 == scrResponse.getInteger("code")) {
                    JSONObject resultObject = scrResponse.getJSONObject("data");
                    if (200 == resultObject.getInteger("code")) {
                        System.out.println(resultObject.getString("personId"));
                    } else {
                        System.out.println("task process fail:" + resultObject.getInteger("
code"));
                    }
                } else {
```

```
                } else {
                    System.out.println("detect not success. code:" + scrResponse.getInteger
("code"));
                }
            } else {
                System.out.println("response not success. status:" + httpResponse.getStatus
());
            }
        } catch (ServerException e) {
            e.printStackTrace();
        } catch (ClientException e) {
            e.printStackTrace();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

2.6.3.4. 获取个体信息

本文介绍了如何使用Java SDK获取个体信息。

功能描述

关于参数的详细说明，请参见查询个体信息API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Java依赖。关于安装Java依赖的具体操作，请参见安装Java依赖。

  > **说明** 请一定按照安装Java依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

获取个体信息代码示例

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.green.model.v20180509.GetPersonRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
/**
 * 获取个体信息
 *
 * @author
```

```
 * @date 2018/06/25
 */
public class FaceGetPersonRequestSample  {
    public static void main(String[] args) throws Exception {
        // 请替换成您的AccessKey ID、AccessKey Secret。
        IClientProfile profile = DefaultProfile.getProfile("cn-shanghai", "您的AccessKey ID"
, "您的AccessKey Secret");
        DefaultProfile.addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com"
);
        IAcsClient client = new DefaultAcsClient(profile);
        GetPersonRequest getPersonRequest = new GetPersonRequest();
        getPersonRequest.setAcceptFormat(FormatType.JSON); // 指定API返回格式。
        getPersonRequest.setMethod(com.aliyuncs.http.MethodType.POST); // 指定请求方法。
        getPersonRequest.setEncoding("utf-8");
        JSONObject data = new JSONObject();
        /**
         * personId: 用户自定义个体ID，必填。
         */
        data.put("personId", "personId_test_3");
        getPersonRequest.setHttpContent(data.toJSONString().getBytes("UTF-8"), "UTF-8", For
matType.JSON);
        /**
         * 请务必设置超时时间。
         */
        getPersonRequest.setConnectTimeout(3000);
        getPersonRequest.setReadTimeout(6000);
        try {
            HttpResponse httpResponse = client.doAction(getPersonRequest);
            if (httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttpCo
ntent(), "UTF-8"));
                System.out.println(JSON.toJSONString(scrResponse, true));
                if (200 == scrResponse.getInteger("code")) {
                    JSONObject resultObject = scrResponse.getJSONObject("data");
                    if (200 == resultObject.getInteger("code")) {
                        System.out.println(resultObject.getString("personId"));
                    } else {
                        System.out.println("task process fail:" + resultObject.getInteger("
code"));
                    }
                } else {
                    System.out.println("detect not success. code:" + scrResponse.getInteger
("code"));
                }
            } else {
                System.out.println("response not success. status:" + httpResponse.getStatus
());
            }
        } catch (ServerException e) {
            e.printStackTrace();
        } catch (ClientException e) {
            e.printStackTrace();
        } catch (Exception e) {
            e.printStackTrace();
```

```
        }
    }
}
```

2.6.3.5. 查询人脸信息

本文介绍了如何使用Java SDK获取人脸信息。

功能描述

根据个体ID和人脸ID查询当前个体的人脸ID以及人脸图片路径等信息。关于查询人脸的具体参数说明，请参见查询人脸图片API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Java依赖。关于安装Java依赖的具体操作，请参见安装Java依赖。

  > **说明** 请一定按照安装Java依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

查询人脸信息代码示例

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.green.model.v20180509.GetFacesRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import java.util.Arrays;
public class FaceGetRequestSample {
    public static void main(String[] args) throws Exception {
        // 请替换成您的AccessKey ID和AccessKey Secret。
        IClientProfile profile = DefaultProfile.getProfile("cn-shanghai", "yourAccessKeyId"
, "yourAccessKeySecret");
        DefaultProfile.addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com"
);
        IAcsClient client = new DefaultAcsClient(profile);
        GetFacesRequest getFacesRequest=new GetFacesRequest();
        // 指定API返回格式。
        getFacesRequest.setAcceptFormat(FormatType.JSON);
        // 指定请求方法。
        getFacesRequest.setMethod(com.aliyuncs.http.MethodType.POST);
        getFacesRequest.setEncoding("utf-8");
        JSONObject jsonObject=new JSONObject();
        jsonObject.put("personId","personId_test_3");
```

```
        jsonObject.put("faceIds", Arrays.asList("203452206484505786"));
        getFacesRequest.setHttpContent(jsonObject.toJSONString().getBytes("UTF-8"), "UTF-8"
, FormatType.JSON);
        // 请务必设置超时时间（单位ms）。
        getFacesRequest.setConnectTimeout(3000);
        getFacesRequest.setReadTimeout(6000);
        try {
            HttpResponse httpResponse = client.doAction(getFacesRequest);
            if (httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttpCo
ntent(), "UTF-8"));
                System.out.println(JSON.toJSONString(scrResponse, true));
                if (200 == scrResponse.getInteger("code")) {
                    JSONObject resultObject = scrResponse.getJSONObject("data");
                    if (200 == resultObject.getInteger("code")) {
                        System.out.println(resultObject.getString("personId"));
                    } else {
                        System.out.println("task process fail:" + resultObject.getInteger("
code"));
                    }
                } else {
                    System.out.println("detect not success. code:" + scrResponse.getInteger
("code"));
                }
            } else {
                System.out.println("response not success. status:" + httpResponse.getStatus
());
            }
        } catch (ServerException e) {
            e.printStackTrace();
        } catch (ClientException e) {
            e.printStackTrace();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

2.6.3.6. 删除个体

本文介绍了如何使用Java SDK删除个体信息。

功能描述

删除个体时，个体对应的图片以及信息均会被删除。关于参数的详细说明，请参见删除个体API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Java依赖。关于安装Java依赖的具体操作，请参见安装Java依赖。

> **说明** 请一定按照安装Java依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

删除个体代码示例

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.green.model.v20180509.DeletePersonRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
/**
 * 删除个体。
 *
 * @author
 * @date 2018/06/25
 */
public class FaceDeletePersonRequestSample  {
    public static void main(String[] args) throws Exception {
        //请替换成您的AccessKey ID、AccessKey Secret。
        IClientProfile profile = DefaultProfile.getProfile("cn-shanghai", "您的AccessKey ID"
, "您的AccessKey Secret");
        DefaultProfile.addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com"
);
        IAcsClient client = new DefaultAcsClient(profile);
        DeletePersonRequest request = new DeletePersonRequest();
        request.setAcceptFormat(FormatType.JSON); // 指定API返回格式。
        request.setMethod(com.aliyuncs.http.MethodType.POST); // 指定请求方法。
        request.setEncoding("utf-8");
        JSONObject data = new JSONObject();
        /**
         * personId: 用户自定义个体ID，必填。
         */
        data.put("personId", "personId_test_1");
        request.setHttpContent(data.toJSONString().getBytes("UTF-8"), "UTF-8", FormatType.J
SON);
        /**
         * 请务必设置超时时间。
         */
        request.setConnectTimeout(3000);
        request.setReadTimeout(6000);
        try {
            HttpResponse httpResponse = client.doAction(request);
            if (httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttpCo
ntent(), "UTF-8"));
                System.out.println(JSON.toJSONString(scrResponse, true));
```

```
                if (200 == scrResponse.getInteger("code")) {
                    JSONObject resultObject = scrResponse.getJSONObject("data");
                    if (200 == resultObject.getInteger("code")) {
                        System.out.println(resultObject.getString("personId"));
                    } else {
                        System.out.println("task process fail:" + resultObject.getInteger("
code"));
                    }
                } else {
                    System.out.println("detect not success. code:" + scrResponse.getInteger
("code"));
                }
            } else {
                System.out.println("response not success. status:" + httpResponse.getStatus
());
            }
        } catch (ServerException e) {
            e.printStackTrace();
        } catch (ClientException e) {
            e.printStackTrace();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

2.6.3.7. 新增人脸

本文介绍了如何使用Java SDK新增个体的人脸图片。

功能描述

为个体新增一组人脸图片，一个个体最多允许包含20张人脸图片。关于参数的说明，请参见新增人脸API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Java依赖。关于安装Java依赖的具体操作，请参见安装Java依赖。

  > **说明** 请一定按照安装Java依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

新增人脸图片（URL）代码示例

```
import java.util.Arrays;
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
```

```
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.green.model.v20180509.AddFacesRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
/**
 * 增加人脸信息，会返回增加成功的人脸ID。
 *
 * @author
 * @date 2018/06/25
 */
public class FaceAddFaceRequestSample  {
    public static void main(String[] args) throws Exception {
        //请替换成您的AccessKey ID、AccessKey Secret。
        IClientProfile profile = DefaultProfile.getProfile("cn-shanghai", "您的AccessKey ID"
, "您的AccessKey Secret");
        DefaultProfile.addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com"
);
        IAcsClient client = new DefaultAcsClient(profile);
        AddFacesRequest addFacesRequest = new AddFacesRequest();
        addFacesRequest.setAcceptFormat(FormatType.JSON); // 指定API返回格式。
        addFacesRequest.setMethod(com.aliyuncs.http.MethodType.POST); // 指定请求方法。
        addFacesRequest.setEncoding("utf-8");
        JSONObject data = new JSONObject();
        /**
         * personId: 用户自定义个体ID，必填。
         * urls: 图片URL。
         */
        data.put("personId", "personId_test_3");
        data.put("urls", Arrays.asList("https://example.com/tfs/xxx-550-407.jpg"));
        addFacesRequest.setHttpContent(data.toJSONString().getBytes("UTF-8"), "UTF-8", Form
atType.JSON);
        /**
         * 请务必设置超时时间。
         */
        addFacesRequest.setConnectTimeout(3000);
        addFacesRequest.setReadTimeout(6000);
        try {
            HttpResponse httpResponse = client.doAction(addFacesRequest);
            if (httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttpCo
ntent(), "UTF-8"));
                System.out.println(JSON.toJSONString(scrResponse, true));
                if (200 == scrResponse.getInteger("code")) {
                    JSONObject resultObject = scrResponse.getJSONObject("data");
                    if (200 == resultObject.getInteger("code")) {
                        System.out.println(resultObject.getString("personId"));
                    } else {
                        System.out.println("task process fail:" + resultObject.getInteger("
code"));
                    }
                } else {
                    System.out.println("detect not success. code:" + scrResponse.getInteger
```

```
("code"));
                }
            } else {
                System.out.println("response not success. status:" + httpResponse.getStatus
());
            }
        } catch (ServerException e) {
            e.printStackTrace();
        } catch (ClientException e) {
            e.printStackTrace();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

新增人脸图片（本地文件）代码示例

```
import java.util.Arrays;
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.green.extension.uploader.ClientUploader;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.green.model.v20180509.AddFacesRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
/**
 * 增加人脸信息，会返回增加成功的人脸ID。
 *
 * @author
 * @date 2018/06/25
 */
public class FaceAddFaceRequestSample  {
    public static void main(String[] args) throws Exception {
        // 请替换成您的AccessKey ID、AccessKey Secret。
        IClientProfile profile = DefaultProfile.getProfile("cn-shanghai", "您的AccessKey ID"
, "您的AccessKey Secret");
        DefaultProfile.addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com"
);
        IAcsClient client = new DefaultAcsClient(profile);
        AddFacesRequest addFacesRequest = new AddFacesRequest();
        addFacesRequest.setAcceptFormat(FormatType.JSON); // 指定API返回格式。
        addFacesRequest.setMethod(com.aliyuncs.http.MethodType.POST); // 指定请求方法。
        addFacesRequest.setEncoding("utf-8");
        JSONObject data = new JSONObject();
        String url = null;
        ClientUploader clientUploader = ClientUploader.getImageClientUploader(profile, fals
e);
```

```
        try{
            url = clientUploader.uploadFile("d:/test.jpg");
        }catch (Exception e){
            e.printStackTrace();
        }
        /**
         * personId: 用户自定义个体ID，必填。
         * urls: 图片URL。
         */
        data.put("personId", "personId_test_3");
        data.put("urls", Arrays.asList(url));
        addFacesRequest.setHttpContent(data.toJSONString().getBytes("UTF-8"), "UTF-8", Form
atType.JSON);
        /**
         * 请务必设置超时时间。
         */
        addFacesRequest.setConnectTimeout(3000);
        addFacesRequest.setReadTimeout(6000);
        try {
            HttpResponse httpResponse = client.doAction(addFacesRequest);
            if (httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttpCo
ntent(), "UTF-8"));
                System.out.println(JSON.toJSONString(scrResponse, true));
                if (200 == scrResponse.getInteger("code")) {
                    JSONObject resultObject = scrResponse.getJSONObject("data");
                    if (200 == resultObject.getInteger("code")) {
                        System.out.println(resultObject.getString("personId"));
                    } else {
                        System.out.println("task process fail:" + resultObject.getInteger("
code"));
                    }
                } else {
                    System.out.println("detect not success. code:" + scrResponse.getInteger
("code"));
                }
            } else {
                System.out.println("response not success. status:" + httpResponse.getStatus
());
            }
        } catch (ServerException e) {
            e.printStackTrace();
        } catch (ClientException e) {
            e.printStackTrace();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

2.6.3.8. 删除人脸

本文介绍了如何使用Java SDK删除个体的人脸图片。

功能描述

删除人脸图片时，必须指定要删除的图片ID以及对应的个体ID信息。关于参数的详细说明，请参见删除人脸API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

功能描述

删除人脸图片时，必须指定要删除的图片ID以及对应的个体ID信息。关于参数说明，请参见删除人脸API接口描述文档。

前提条件

- 安装Java依赖。关于安装Java依赖的具体操作，请参见安装Java依赖。

  > **说明** 请一定按照安装Java依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

删除人脸代码示例

```
import java.util.Arrays;
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.green.model.v20180509.DeleteFacesRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
public class FaceDeleteFaceRequestSample {
    public static void main(String[] args) throws Exception {
        // 请替换成您的AccessKey ID、AccessKey Secret。
        IClientProfile profile = DefaultProfile.getProfile("cn-shanghai", "您的AccessKey ID"
, "您的AccessKey Secret");
        DefaultProfile.addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com"
);
        IAcsClient client = new DefaultAcsClient(profile);
        DeleteFacesRequest deleteFacesRequest = new DeleteFacesRequest();
        deleteFacesRequest.setAcceptFormat(FormatType.JSON); // 指定API返回格式。
        deleteFacesRequest.setMethod(com.aliyuncs.http.MethodType.POST); // 指定请求方法。
        deleteFacesRequest.setEncoding("utf-8");
        JSONObject data = new JSONObject();
        /**
         * personId: 用户自定义个体ID，必填。
         * faceIds: 已添加的人脸ID。
         */
        data.put("personId", "personId_test_3");
```

```
        data.put("faceIds", Arrays.asList("31820666292926465"));
        deleteFacesRequest.setHttpContent(data.toJSONString().getBytes("UTF-8"), "UTF-8", F
ormatType.JSON);
        /**
         * 请务必设置超时时间。
         */
        deleteFacesRequest.setConnectTimeout(3000);
        deleteFacesRequest.setReadTimeout(6000);
        try {
            HttpResponse httpResponse = client.doAction(deleteFacesRequest);
            if (httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttpCo
ntent(), "UTF-8"));
                System.out.println(JSON.toJSONString(scrResponse, true));
                if (200 == scrResponse.getInteger("code")) {
                    JSONObject resultObject = scrResponse.getJSONObject("data");
                    if (200 == resultObject.getInteger("code")) {
                        System.out.println(resultObject.getString("personId"));
                    } else {
                        System.out.println("task process fail:" + resultObject.getInteger("
code"));
                    }
                } else {
                    System.out.println("detect not success. code:" + scrResponse.getInteger
("code"));
                }
            } else {
                System.out.println("response not success. status:" + httpResponse.getStatus
());
            }
        } catch (ServerException e) {
            e.printStackTrace();
        } catch (ClientException e) {
            e.printStackTrace();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

2.6.3.9. 获取组列表

本文介绍了如何使用Java SDK获取所有人脸分组ID。

功能描述

关于参数的详细说明，请参见查询个体信息API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Java依赖。关于安装Java依赖的具体操作，请参见安装Java依赖。

> **说明** 请一定按照安装Java依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

获取所有分组ID代码示例

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.green.model.v20180509.GetGroupsRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
/**
 * 获取用户下所有的组ID列表。
 *
 * @author
 * @date 2018/06/25
 */
public class FaceGetGroupsRequestSample  {
    public static void main(String[] args) throws Exception {
        // 请替换成您的AccessKey ID、AccessKey Secret。
        IClientProfile profile = DefaultProfile.getProfile("cn-shanghai", "您的AccessKey ID"
, "您的AccessKey Secret");
        DefaultProfile.addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com"
);
        IAcsClient client = new DefaultAcsClient(profile);
        GetGroupsRequest request = new GetGroupsRequest();
        request.setAcceptFormat(FormatType.JSON); // 指定API返回格式。
        request.setMethod(com.aliyuncs.http.MethodType.POST); // 指定请求方法。
        request.setEncoding("utf-8");
        JSONObject data = new JSONObject();
        request.setHttpContent(data.toJSONString().getBytes("UTF-8"), "UTF-8", FormatType.J
SON);
        /**
         * 请务必设置超时时间。
         */
        request.setConnectTimeout(3000);
        request.setReadTimeout(6000);
        try {
            HttpResponse httpResponse = client.doAction(request);
            if (httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttpCo
ntent(), "UTF-8"));
                System.out.println(JSON.toJSONString(scrResponse, true));
                if (200 == scrResponse.getInteger("code")) {
                    JSONObject resultObject = scrResponse.getJSONObject("data");
                    if (200 == resultObject.getInteger("code")) {
                        System.out.println(resultObject.getString("groupIds"));
```

```
                    } else {
                        System.out.println("task process fail:" + resultObject.getInteger("
code"));
                    }
                } else {
                    System.out.println("detect not success. code:" + scrResponse.getInteger
("code"));
                }
            } else {
                System.out.println("response not success. status:" + httpResponse.getStatus
());
            }
        } catch (ServerException e) {
            e.printStackTrace();
        } catch (ClientException e) {
            e.printStackTrace();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

2.6.3.10. 获取个体列表

本文介绍了如何使用Java SDK获取个体列表信息。

功能描述

每个个体必须属于一个分组，在调用该接口时指定某个分组，则返回该分组下的所有个体ID列表。关于参数的详细说明，请参见查询个体列表API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Java依赖。关于安装Java依赖的具体操作，请参见安装Java依赖。

  > **说明** 请一定按照安装Java依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

获取个体列表代码示例

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.green.model.v20180509.GetPersonsRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.profile.DefaultProfile;
```

```
import com.aliyuncs.profile.IClientProfile;
/**
 * 获取指定组下的个体Id列表。
 *
 * @author wangxuan
 * @date 2018/06/25
 */
public class FaceGetPersonsRequestSample {
    public static void main(String[] args) throws Exception {
        //请替换成您的AccessKey ID、AccessKey Secret。
        IClientProfile profile = DefaultProfile.getProfile("cn-shanghai", "您的AccessKey ID"
, "您的AccessKey Secret");
        DefaultProfile.addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com"
);
        IAcsClient client = new DefaultAcsClient(profile);
        GetPersonsRequest request = new GetPersonsRequest();
        request.setAcceptFormat(FormatType.JSON); // 指定API返回格式。
        request.setMethod(com.aliyuncs.http.MethodType.POST); // 指定请求方法。
        request.setEncoding("utf-8");
        JSONObject data = new JSONObject();
        /**
         * groupId: 用户自定义组ID，必填。
         */
        data.put("groupId", "java_sdk_test_group");
        request.setHttpContent(data.toJSONString().getBytes("UTF-8"), "UTF-8", FormatType.J
SON);
        /**
         * 请务必设置超时时间。
         */
        request.setConnectTimeout(3000);
        request.setReadTimeout(6000);
        try {
            HttpResponse httpResponse = client.doAction(request);
            if (httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttpCo
ntent(), "UTF-8"));
                System.out.println(JSON.toJSONString(scrResponse, true));
                if (200 == scrResponse.getInteger("code")) {
                    JSONObject resultObject = scrResponse.getJSONObject("data");
                    if (200 == resultObject.getInteger("code")) {
                        System.out.println(resultObject.getString("personIds"));
                    } else {
                        System.out.println("task process fail:" + resultObject.getInteger("
code"));
                    }
                } else {
                    System.out.println("detect not success. code:" + scrResponse.getInteger
("code"));
                }
            } else {
                System.out.println("response not success. status:" + httpResponse.getStatus
());
            }
        } catch (ServerException e) {
```

```
            e.printStackTrace();
        } catch (ClientException e) {
            e.printStackTrace();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

2.6.3.11. 添加个体到分组

本文介绍了如何使用Java SDK添加个体到指定分组。

功能描述

将一个个体添加到一个或者多个个体组。关于参数的详细说明，请参见添加个体到分组API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Java依赖。关于安装Java依赖的具体操作，请参见安装Java依赖。

  > **说明** 请一定按照安装Java依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

添加个体到分组代码示例

```
import java.util.Arrays;
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.green.model.v20180509.AddPersonRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
/**
 * 增加个体到指定分组。
 * @author
 * @date 2018/06/25
 */
public class FaceAddPersonRequestSample {
    public static void main(String[] args) throws Exception {
        //请替换成您的AccessKey ID、AccessKey Secret。
        IClientProfile profile = DefaultProfile.getProfile("cn-shanghai", "您的AccessKey ID"
, "您的AccessKey Secret");
        DefaultProfile.addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com"
```

```
);
        IAcsClient client = new DefaultAcsClient(profile);
        AddGroupsRequest request = new AddGroupsRequest();
        request.setAcceptFormat(FormatType.JSON); // 指定API返回格式。
        request.setMethod(com.aliyuncs.http.MethodType.POST); // 指定请求方法。
        request.setEncoding("utf-8");
        JSONObject data = new JSONObject();
        /**
         * personId: 用户自定义个体ID，必填。
         * groupIds: 用户自定义组ID列表，必填。
         */
        data.put("personId", "personId_test_3");
        data.put("groupIds", Arrays.asList("groupId_1"));
        request.setHttpContent(data.toJSONString().getBytes("UTF-8"), "UTF-8", FormatType.J
SON);
        /**
         * 请务必设置超时时间。
         */
        request.setConnectTimeout(3000);
        request.setReadTimeout(6000);
        try {
            HttpResponse httpResponse = client.doAction(request);
            if (httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttpCo
ntent(), "UTF-8"));
                System.out.println(JSON.toJSONString(scrResponse, true));
                if (200 == scrResponse.getInteger("code")) {
                    JSONObject resultObject = scrResponse.getJSONObject("data");
                    if (200 == resultObject.getInteger("code")) {
                        System.out.println(resultObject.getString("personId"));
                    } else {
                        System.out.println("task process fail:" + resultObject.getInteger("
code"));
                    }
                } else {
                    System.out.println("detect not success. code:" + scrResponse.getInteger
("code"));
                }
            } else {
                System.out.println("response not success. status:" + httpResponse.getStatus
());
            }
        } catch (ServerException e) {
            e.printStackTrace();
        } catch (ClientException e) {
            e.printStackTrace();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

2.6.3.12. 删除分组中个体

本文介绍了如何使用Java SDK从指定分组中删除个体信息。

功能描述

删除分组中个体不会删除个体对应的信息以及图片，只是解除个体与该分组的绑定关系。关于参数的详细说明，请参见删除分组中的个体API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Java依赖。关于安装Java依赖的具体操作，请参见安装Java依赖。

  > **说明** 请一定按照安装Java依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

从指定分组删除个体代码示例

```
import java.util.Arrays;
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.green.model.v20180509.DeleteGroupsRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
/**
 * 将个体从指定组中移除
 *
 * @author wangxuan
 * @date 2018/06/25
 */
public class FaceDeleteGroupsRequestSample {
    public static void main(String[] args) throws Exception {
        //请替换成您的AccessKey ID、AccessKey Secret。
        IClientProfile profile = DefaultProfile.getProfile("cn-shanghai", "您的AccessKey ID"
, "您的AccessKey Secret");
        DefaultProfile.addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com"
);
        IAcsClient client = new DefaultAcsClient(profile);
        DeleteGroupsRequest request = new DeleteGroupsRequest();
        request.setAcceptFormat(FormatType.JSON); // 指定API返回格式。
        request.setMethod(com.aliyuncs.http.MethodType.POST); // 指定请求方法。
        request.setEncoding("utf-8");
        JSONObject data = new JSONObject();
        /**
```

```
         * personId: 用户自定义个体ID，必填。
         * groupIds: 用户自定义组ID列表，必填。
         */
        data.put("personId", "personId_test_3");
        data.put("groupIds", Arrays.asList("groupId_1"));
        request.setHttpContent(data.toJSONString().getBytes("UTF-8"), "UTF-8", FormatType.J
SON);
        /**
         * 请务必设置超时时间。
         */
        request.setConnectTimeout(3000);
        request.setReadTimeout(6000);
        try {
            HttpResponse httpResponse = client.doAction(request);
            if (httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttpCo
ntent(), "UTF-8"));
                System.out.println(JSON.toJSONString(scrResponse, true));
                if (200 == scrResponse.getInteger("code")) {
                    JSONObject resultObject = scrResponse.getJSONObject("data");
                    if (200 == resultObject.getInteger("code")) {
                        System.out.println(resultObject.getString("personId"));
                    } else {
                        System.out.println("task process fail:" + resultObject.getInteger("
code"));
                    }
                } else {
                    System.out.println("detect not success. code:" + scrResponse.getInteger
("code"));
                }
            } else {
                System.out.println("response not success. status:" + httpResponse.getStatus
());
            }
        } catch (ServerException e) {
            e.printStackTrace();
        } catch (ClientException e) {
            e.printStackTrace();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

2.7. 其他

2.7.1. 相似图检索

2.7.1.1. 创建相似图样本库

本文介绍了如何使用Java SDK，创建相似图样本库。

功能描述

通过该接口创建相似图样本图库。关于参数的详细说明，请参见创建图库API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Java依赖。关于安装Java依赖的具体操作，请参见安装Java依赖。

  > **说明** 请一定按照安装Java依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

创建相似图样本任务

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.green.model.v20180509.AddSimilarityLibraryRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.http.MethodType;
import com.aliyuncs.http.ProtocolType;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
public class AddSimilarityLibSample {
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile
                .getProfile("cn-shanghai", "您的AccessKey ID", "您的AccessKey Secret");
        DefaultProfile
                .addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        AddSimilarityLibraryRequest request = new AddSimilarityLibraryRequest();
        // 指定API返回格式、请求方法。
        request.setAcceptFormat(FormatType.JSON);
        request.setMethod(MethodType.POST);
        request.setEncoding("utf-8");
        request.setProtocol(ProtocolType.HTTP);
        JSONObject httpBody = new JSONObject();
        httpBody.put("name", "相似图样本图库名称");
        request.setHttpContent(org.apache.commons.codec.binary.StringUtils.getBytesUtf8(htt
pBody.toJSONString()),
                "UTF-8", FormatType.JSON);
        /**
         * 请设置超时时间，服务端全链路处理超时时间为10秒，请做相应设置。
         * 如果您设置的ReadTimeout小于服务端处理的时间，程序中会获得一个ReadTimeout异常。
         */
        request.setConnectTimeout(3000);
        request.setReadTimeout(10000);
        HttpResponse httpResponse = null;
        try {
            httpResponse = client.doAction(request);
```

```
            // 服务端接收到请求，完成处理后返回的结果。
            if (httpResponse != null && httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(org.apache.commons.codec.binary.S
tringUtils.newStringUtf8(httpResponse.getHttpContent()));
                System.out.println(JSON.toJSONString(scrResponse, true));
                int requestCode = scrResponse.getIntValue("code");
                JSONObject data = scrResponse.getJSONObject("data");
                if (200 == requestCode) {
                    // 创建图库的时间戳。
                    int createTime = data.getIntValue("createTime");
                    // 图库中的图片样本的数量。
                    int imageCount = data.getIntValue("imageCount");
                    // 图库名称。
                    String name = data.getString("name");
                    // 图库所属区域。
                    String region = data.getString("region");
                } else {
                    /**
                     * 表明请求整体处理失败，原因视具体的情况详细分析。
                     */
                    System.out.println("the whole request failed. response:" + JSON.toJSONS
tring(scrResponse));
                }
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

2.7.1.2. 相似图检索

本文介绍了如何使用Java SDK，对指定的图片进行相似图检索。

功能描述

相似图检索帮助您从图库中检索出与给定的图片相似的若干张图片。关于参数的详细说明，请参见相似图检索API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Java依赖。关于安装Java依赖的具体操作，请参见安装Java依赖。

  > **说明** 请一定按照安装Java依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

传入URL示例代码

```
import com.alibaba.fastjson.JSON;
```

```
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.green.model.v20180509.ImageSyncScanRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.http.MethodType;
import com.aliyuncs.http.ProtocolType;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import java.util.*;
public class Main {
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile
            .getProfile("cn-shanghai", "请填写您的AccessKey ID", "请填写您的AccessKey Secret")
;
        DefaultProfile
            .addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        // 指定API返回格式、请求方法。
        ImageSyncScanRequest imageSyncScanRequest = new ImageSyncScanRequest();
        imageSyncScanRequest.setAcceptFormat(FormatType.JSON);
        imageSyncScanRequest.setMethod(MethodType.POST);
        imageSyncScanRequest.setEncoding("utf-8");
        imageSyncScanRequest.setProtocol(ProtocolType.HTTP);
        JSONObject httpBody = new JSONObject();
        /**
         * 设置要检测的场景，计费是按照该处传递的场景进行。
         * similarity: similarity表示进行相似图检索。
         */
        httpBody.put("scenes", Arrays.asList("similarity"));
        /**
         * 设置待检测图片，一张图片一个Task。
         * 多张图片同时检测时，处理的时间由最后一个处理完的图片决定。
         * 通常情况下批量检测的平均RT比单张检测的要长，一次批量提交的图片数越多，RT被拉长的概率越高。
         * 这里以单张图片检测作为示例，如果是批量图片检测，请自行构建多个Task。
         */
        JSONObject task = new JSONObject();
        task.put("dataId", UUID.randomUUID().toString());
        // 设置待检索的图片链接。
        task.put("url", "https://example.com/tfs/xxx-600-600.jpg");
        httpBody.put("tasks", Arrays.asList(task));
        imageSyncScanRequest.setHttpContent(org.apache.commons.codec.binary.StringUtils.get
BytesUtf8(httpBody.toJSONString()),
            "UTF-8", FormatType.JSON);
        /**
         * 请设置超时时间，服务端全链路处理超时时间为10秒，请做相应设置。
         * 如果您设置的ReadTimeout小于服务端处理的时间，程序中会获得一个ReadTimeout异常。
         */
        imageSyncScanRequest.setConnectTimeout(3000);
        imageSyncScanRequest.setReadTimeout(10000);
        HttpResponse httpResponse = null;
        try {
```

```
            httpResponse = client.doAction(imageSyncScanRequest);
        } catch (Exception e) {
            e.printStackTrace();
        }
        // 服务端接收到请求，并完成处理返回的结果。
        if (httpResponse != null && httpResponse.isSuccess()) {
            JSONObject scrResponse = JSON.parseObject(org.apache.commons.codec.binary.Strin
gUtils.newStringUtf8(httpResponse.getHttpContent()));
            System.out.println(JSON.toJSONString(scrResponse, true));
            int requestCode = scrResponse.getIntValue("code");
            // 每一张图片的检测结果。
            JSONArray taskResults = scrResponse.getJSONArray("data");
            if (200 == requestCode) {
                for (Object taskResult : taskResults) {
                    // 单张图片的处理结果。
                    int taskCode = ((JSONObject) taskResult).getIntValue("code");
                    // 图片要检测的场景的处理结果，如果是多个场景，则会有每个场景的结果。
                    JSONArray sceneResults = ((JSONObject) taskResult).getJSONArray("result
s");
                    if (200 == taskCode) {
                        for (Object sceneResult : sceneResults) {
                            String scene = ((JSONObject) sceneResult).getString("scene");
                            String suggestion = ((JSONObject) sceneResult).getString("sugge
stion");
                            // 根据Suggetion做相关处理。
                            System.out.println("suggestion = [" + suggestion + "]");
                        }
                    } else {
                        // 单张图片处理失败，原因是具体的情况详细分析。
                        System.out.println("task process fail. task response:" + JSON.toJSO
NString(taskResult));
                    }
                }
            } else {
                /**
                 * 表明请求整体处理失败，原因视具体的情况详细分析。
                 */
                System.out.println("the whole image scan request failed. response:" + JSON.
toJSONString(scrResponse));
            }
        }
    }
}
```

传入本地文件示例代码

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.green.extension.uploader.ClientUploader;
import com.aliyuncs.green.model.v20180509.ImageSyncScanRequest;
import com.aliyuncs.http.FormatType;
```

```
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.http.MethodType;
import com.aliyuncs.http.ProtocolType;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import java.util.Arrays;
import java.util.UUID;
public class SimilarityImageSyncScanLocalSample {
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile
                .getProfile("cn-shanghai", "请填写您的AccessKey ID", "请填写您的AccessKey Secr
et");
        DefaultProfile
                .addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        // 指定API返回格式、请求方法。
        ImageSyncScanRequest imageSyncScanRequest = new ImageSyncScanRequest();
        imageSyncScanRequest.setAcceptFormat(FormatType.JSON);
        imageSyncScanRequest.setMethod(MethodType.POST);
        imageSyncScanRequest.setEncoding("utf-8");
        imageSyncScanRequest.setProtocol(ProtocolType.HTTP);
        JSONObject httpBody = new JSONObject();
        /**
         * 设置要检测的场景，计费是按照该处传递的场景进行。
         * similarity: similarity表示进行相似图检索。
         */
        httpBody.put("scenes", Arrays.asList("similarity"));
        /**
         * 设置待检测图片，一张图片一个Task。
         * 多张图片同时检测时，处理的时间由最后一个处理完的图片决定。
         * 通常情况下批量检测的平均RT比单张检测的要长，一次批量提交的图片数越多，RT被拉长的概率越高。
         * 这里以单张图片检测作为示例，如果是批量图片检测，请自行构建多个Task。
         */
        JSONObject task = new JSONObject();
        task.put("dataId", UUID.randomUUID().toString());
        ClientUploader uploader = ClientUploader.getImageClientUploader(profile, false);
        String url = null;
        try {
            url = uploader.uploadFile("d:/test.jpg");
        } catch (Exception e) {
            e.printStackTrace();
        }
        // 设置待检索的图片链接。
        task.put("url", url);
        httpBody.put("tasks", Arrays.asList(task));
        imageSyncScanRequest.setHttpContent(org.apache.commons.codec.binary.StringUtils.get
BytesUtf8(httpBody.toJSONString()),
                "UTF-8", FormatType.JSON);
        /**
         * 请设置超时时间，服务端全链路处理超时时间为10秒，请做相应设置。
         * 如果您设置的ReadTimeout小于服务端处理的时间，程序中会获得一个ReadTimeout异常。
         */
        imageSyncScanRequest.setConnectTimeout(3000);
        imageSyncScanRequest.setReadTimeout(10000);
```

```
        HttpResponse httpResponse = null;
        try {
            httpResponse = client.doAction(imageSyncScanRequest);
        } catch (Exception e) {
            e.printStackTrace();
        }
        // 服务端接收到请求，并完成处理返回的结果。
        if (httpResponse != null && httpResponse.isSuccess()) {
            JSONObject scrResponse = JSON.parseObject(org.apache.commons.codec.binary.Strin
gUtils.newStringUtf8(httpResponse.getHttpContent()));
            System.out.println(JSON.toJSONString(scrResponse, true));
            int requestCode = scrResponse.getIntValue("code");
            // 每一张图片的检测结果。
            JSONArray taskResults = scrResponse.getJSONArray("data");
            if (200 == requestCode) {
                for (Object taskResult : taskResults) {
                    // 单张图片的处理结果。
                    int taskCode = ((JSONObject) taskResult).getIntValue("code");
                    // 图片要检测的场景的处理结果，如果是多个场景，则会有每个场景的结果。
                    JSONArray sceneResults = ((JSONObject) taskResult).getJSONArray("result
s");
                    if (200 == taskCode) {
                        for (Object sceneResult : sceneResults) {
                            String scene = ((JSONObject) sceneResult).getString("scene");
                            String suggestion = ((JSONObject) sceneResult).getString("sugge
stion");
                            // 根据Suggetion做相关处理。
                            System.out.println("suggestion = [" + suggestion + "]");
                        }
                    } else {
                        // 单张图片处理失败，原因是具体的情况详细分析。
                        System.out.println("task process fail. task response:" + JSON.toJSO
NString(taskResult));
                    }
                }
            } else {
                /**
                 * 表明请求整体处理失败，原因视具体的情况详细分析。
                 */
                System.out.println("the whole image scan request failed. response:" + JSON.
toJSONString(scrResponse));
            }
        }
    }
}
```

2.7.1.3. 增加相似图样本

本文介绍了如何使用Java SDK，将指定的相似图图片添加到对应图库。

功能描述

添加相似图样本时，您可以为图片设置标签。如果样本图片在检索时被命中，其标签信息也会被返回。关于参数的详细说明，请参见增加样本图片API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Java依赖。关于安装Java依赖的具体操作，请参见安装Java依赖。

  > **说明** 请一定按照安装Java依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

传入URL示例代码

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.green.model.v20180509.AddSimilarityImageRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.http.MethodType;
import com.aliyuncs.http.ProtocolType;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import java.util.*;
public class Main {
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile
            .getProfile("cn-shanghai", "您的Accesskey ID", "您的Accesskey Secret");
        DefaultProfile
            .addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        AddSimilarityImageRequest addSimilarityImageRequest = new AddSimilarityImageRequest
();
        // 指定API返回格式、请求方法。
        addSimilarityImageRequest.setAcceptFormat(FormatType.JSON);
        addSimilarityImageRequest.setMethod(MethodType.POST);
        addSimilarityImageRequest.setEncoding("utf-8");
        addSimilarityImageRequest.setProtocol(ProtocolType.HTTP);
        JSONObject httpBody = new JSONObject();
        /**
         * 一次请求中可以同时添加多张图片（最大20张/次），
         * 计费按照单张图片添加的单价×添加的图片数量计算。
         */
        JSONObject task = new JSONObject();
        task.put("dataId", UUID.randomUUID().toString());
        // 添加图片。
        task.put("url", "https://example.com/feed/xxx");
        // 添加最多3个自定义标签（非必须）。
        JSONArray tags = new JSONArray();
        tags.add("自定义标签1");
```

```
        tags.add("自定义标签1");
        tags.add("自定义标签2");
        tags.add("自定义标签3");
        task.put("tag", tags);
        httpBody.put("tasks", Arrays.asList(task));
        addSimilarityImageRequest.setHttpContent(org.apache.commons.codec.binary.StringUtil
s.getBytesUtf8(httpBody.toJSONString()),
            "UTF-8", FormatType.JSON);
        /**
         * 请设置超时时间，服务端全链路处理超时时间为10秒，请做相应设置。
         * 如果您设置的ReadTimeout小于服务端处理的时间，程序中会获得一个ReadTimeout异常。
         */
        addSimilarityImageRequest.setConnectTimeout(3000);
        addSimilarityImageRequest.setReadTimeout(10000);
        HttpResponse httpResponse = null;
        try {
            httpResponse = client.doAction(addSimilarityImageRequest);
        } catch (Exception e) {
            e.printStackTrace();
        }
        // 服务端接收到请求，并完成处理返回的结果。
        if (httpResponse != null && httpResponse.isSuccess()) {
            JSONObject scrResponse = JSON.parseObject(org.apache.commons.codec.binary.Strin
gUtils.newStringUtf8(httpResponse.getHttpContent()));
            System.out.println(JSON.toJSONString(scrResponse, true));
            int requestCode = scrResponse.getIntValue("code");
            // 每一张图片的检测结果。
            JSONArray taskResults = scrResponse.getJSONArray("data");
            if (200 == requestCode) {
                for (Object taskResult : taskResults) {
                    // 单张图片的处理结果。
                    int taskCode = ((JSONObject) taskResult).getIntValue("code");
                      if (200 == taskCode) {
                        System.out.println("成功添加1张图片 ");
                    } else {
                        // 单张图片处理失败，原因是具体的情况详细分析。
                        System.out.println("task process fail. task response:" + JSON.toJSO
NString(taskResult));
                    }
                }
            } else {
                /**
                 * 表明请求整体处理失败，原因视具体的情况详细分析。
                 */
                System.out.println("the whole image scan request failed. response:" + JSON.
toJSONString(scrResponse));
            }
        }
    }
}
```

传入本地文件示例代码

```
import com.alibaba.fastjson.JSON;
```

```
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.green.extension.uploader.ClientUploader;
import com.aliyuncs.green.model.v20180509.AddSimilarityImageRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.http.MethodType;
import com.aliyuncs.http.ProtocolType;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import java.util.*;
public class Main {
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile
            .getProfile("cn-shanghai", "您的Accesskey ID", "您的Accesskey Secret");
        DefaultProfile
            .addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        AddSimilarityImageRequest addSimilarityImageRequest = new AddSimilarityImageRequest
();
        // 指定API返回格式、请求方法。
        addSimilarityImageRequest.setAcceptFormat(FormatType.JSON);
        addSimilarityImageRequest.setMethod(MethodType.POST);
        addSimilarityImageRequest.setEncoding("utf-8");
        addSimilarityImageRequest.setProtocol(ProtocolType.HTTP);
        JSONObject httpBody = new JSONObject();
        /**
         * 一次请求中可以同时添加多张图片（最大20张/次），
         * 计费按照单张图片添加的单价×添加的图片数量计算。
         */
        JSONObject task = new JSONObject();
        task.put("dataId", UUID.randomUUID().toString());
         /**
         * 如果您要检测的文件存于本地服务器上，可以通过下述代码片生成URL。
         * 再将返回的URL作为图片地址传递到服务端。
         */
        String url = null;
        ClientUploader clientUploader = ClientUploader.getImageClientUploader(profile, fals
e);
        try{
            url = clientUploader.uploadFile("d:/test.jpg");
        }catch (Exception e){
            e.printStackTrace();
        }
        task.put("url", url);
        // 添加最多3个自定义标签（非必须）。
        JSONArray tags = new JSONArray();
        tags.add("自定义标签1");
        tags.add("自定义标签2");
        tags.add("自定义标签3");
        task.put("tag", tags);
        httpBody.put("tasks", Arrays.asList(task));
```

```
        addSimilarityImageRequest.setHttpContent(org.apache.commons.codec.binary.StringUtil
s.getBytesUtf8(httpBody.toJSONString()),
            "UTF-8", FormatType.JSON);
        /**
         * 请设置超时时间，服务端全链路处理超时时间为10秒，请做相应设置。
         * 如果您设置的ReadTimeout小于服务端处理的时间，程序中会获得一个ReadTimeout异常。
         */
        addSimilarityImageRequest.setConnectTimeout(3000);
        addSimilarityImageRequest.setReadTimeout(10000);
        HttpResponse httpResponse = null;
        try {
            httpResponse = client.doAction(addSimilarityImageRequest);
        } catch (Exception e) {
            e.printStackTrace();
        }
        // 服务端接收到请求，并完成处理返回的结果。
        if (httpResponse != null && httpResponse.isSuccess()) {
            JSONObject scrResponse = JSON.parseObject(org.apache.commons.codec.binary.Strin
gUtils.newStringUtf8(httpResponse.getHttpContent()));
            System.out.println(JSON.toJSONString(scrResponse, true));
            int requestCode = scrResponse.getIntValue("code");
            // 每一张图片的检测结果。
            JSONArray taskResults = scrResponse.getJSONArray("data");
            if (200 == requestCode) {
                for (Object taskResult : taskResults) {
                    // 单张图片的处理结果。
                    int taskCode = ((JSONObject) taskResult).getIntValue("code");
                     if (200 == taskCode) {
                        System.out.println("成功添加1张图片 ");
                    } else {
                        // 单张图片处理失败，原因是具体的情况详细分析。
                        System.out.println("task process fail. task response:" + JSON.toJSO
NString(taskResult));
                    }
                }
            } else {
                /**
                 * 表明请求整体处理失败，原因视具体的情况详细分析。
                 */
                System.out.println("the whole image scan request failed. response:" + JSON.
toJSONString(scrResponse));
            }
        }
    }
}
```

2.7.1.4. 查询相似图库列表

本文介绍了如何使用Java SDK，分页查询相似图库及其元素信息。

功能描述

调用该接口分页查询所有相似图库及其元素信息。关于参数的详细说明，请参见查询相似图库列表API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Java依赖。关于安装Java依赖的具体操作，请参见安装Java依赖。

  > **说明** 请一定按照安装Java依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

查询相似图库列表任务

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.green.model.v20180509.ListSimilarityLibrariesRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.http.MethodType;
import com.aliyuncs.http.ProtocolType;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
public class ListSimilarityLibrariesSample {
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile
                .getProfile("cn-shanghai", "您的AccessKey ID", "您的AccessKey Secret");
        DefaultProfile
                .addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        ListSimilarityLibrariesRequest request = new ListSimilarityLibrariesRequest();
        // 指定API返回格式、请求方法。
        request.setAcceptFormat(FormatType.JSON);
        request.setMethod(MethodType.POST);
        request.setEncoding("utf-8");
        request.setProtocol(ProtocolType.HTTP);
        //pageSize：每页大小，取值范围：(0,50]，currentPage：当前页码，取值范围：(0,50]。
        JSONObject httpBody = new JSONObject();
        httpBody.put("pageSize", "5");
        httpBody.put("currentPage", "1");
        request.setHttpContent(org.apache.commons.codec.binary.StringUtils.getBytesUtf8(htt
pBody.toJSONString()),
                "UTF-8", FormatType.JSON);
        /**
         * 请设置超时时间，服务端全链路处理超时时间为10秒，请做相应设置。
         * 如果您设置的ReadTimeout小于服务端处理的时间，程序中会获得一个ReadTimeout异常。
         */
        request.setConnectTimeout(3000);
        request.setReadTimeout(10000);
        HttpResponse httpResponse = null;
        try {
            httpResponse = client.doAction(request);
```

```
            // 服务端接收到请求，完成处理后返回的结果。
            if (httpResponse != null && httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(org.apache.commons.codec.binary.S
tringUtils.newStringUtf8(httpResponse.getHttpContent()));
                System.out.println(JSON.toJSONString(scrResponse, true));
                int requestCode = scrResponse.getIntValue("code");
                if (200 == requestCode) {
                    // 当前页码。
                    int currentPage = scrResponse.getIntValue("currentPage");
                    // 页面大小。
                    int pageSize = scrResponse.getIntValue("pageSize");
                    // 图库总数量。
                    int totalCount = scrResponse.getIntValue("totalCount");
                    // 具体结果列表。
                    JSONArray results = scrResponse.getJSONArray("data");
                    for (Object data : results) {
                        // 单个结果。
                        // 创建图库的时间戳。
                        int createTime = ((JSONObject) data).getIntValue("createTime");
                        // 图库中的图片样本的数量。
                        int imageCount = ((JSONObject) data).getIntValue("imageCount");
                        // 图库名称。
                        String name = ((JSONObject) data).getString("name");
                        // 图库所属区域。
                        String region = ((JSONObject) data).getString("region");
                    }
                } else {
                    /**
                     * 表明请求整体处理失败，原因视具体的情况详细分析。
                     */
                    System.out.println("the whole request failed. response:" + JSON.toJSONS
tring(scrResponse));
                }
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

2.7.1.5. 查询指定相似图库信息

本文介绍了如何使用Java SDK查询指定相似图库信息。

功能描述

调用该接口查询指定相似图库信息。关于参数的详细说明，请参见查询指定相似图库API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Java依赖。关于安装Java依赖的具体操作，请参见安装Java依赖。

  > **说明** 请一定按照安装Java依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

查询指定相似图库任务

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.green.model.v20180509.GetSimilarityLibraryRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.http.MethodType;
import com.aliyuncs.http.ProtocolType;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
public class GetSimilarityLibrarySample {
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile
                .getProfile("cn-shanghai", "您的AccessKey ID", "您的AccessKey Secret");
        DefaultProfile
                .addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        GetSimilarityLibraryRequest request = new GetSimilarityLibraryRequest();
        // 指定API返回格式、请求方法。
        request.setAcceptFormat(FormatType.JSON);
        request.setMethod(MethodType.POST);
        request.setEncoding("utf-8");
        request.setProtocol(ProtocolType.HTTP);
        JSONObject httpBody = new JSONObject();
        httpBody.put("name", "相似图样本图库名称");
        request.setHttpContent(org.apache.commons.codec.binary.StringUtils.getBytesUtf8(htt
pBody.toJSONString()),
                "UTF-8", FormatType.JSON);
        /**
         * 请设置超时时间，服务端全链路处理超时时间为10秒，请做相应设置。
         * 如果您设置的ReadTimeout小于服务端处理的时间，程序中会获得一个ReadTimeout异常。
         */
        request.setConnectTimeout(3000);
        request.setReadTimeout(10000);
        HttpResponse httpResponse = null;
        try {
            httpResponse = client.doAction(request);
            // 服务端接收到请求，完成处理后返回的结果。
            if (httpResponse != null && httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(org.apache.commons.codec.binary.S
tringUtils.newStringUtf8(httpResponse.getHttpContent()));
                System.out.println(JSON.toJSONString(scrResponse, true));
                int requestCode = scrResponse.getIntValue("code");
                JSONObject data = scrResponse.getJSONObject("data");
```

```
                if (200 == requestCode) {
                    // 创建图库的时间戳。
                    int createTime = data.getIntValue("createTime");
                    // 图库中的图片样本的数量。
                    int imageCount = data.getIntValue("imageCount");
                    // 图库名称。
                    String name = data.getString("name");
                    // 图库所属区域。
                    String region = data.getString("region");
                } else {
                    /**
                     * 表明请求整体处理失败，原因视具体的情况详细分析。
                     */
                    System.out.println("the whole request failed. response:" + JSON.toJSONS
tring(scrResponse));
                }
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

2.7.1.6. 查询相似图样本图片列表

本文介绍了如何使用Java SDK，分页查询指定相似图库下的所有样本图片信息。

功能描述

调用该接口分页查询指定相似图库下的所有样本图片信息。关于参数的详细说明，请参见查询样本图片列表API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Java依赖。关于安装Java依赖的具体操作，请参见安装Java依赖。

  > **说明** 请一定按照安装Java依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

查询相似图样本图片列表任务

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.green.model.v20180509.ListSimilarityImagesRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
```

```
import com.aliyuncs.http.MethodType;
import com.aliyuncs.http.ProtocolType;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
public class ListSimilarityImagesSample {
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile
                .getProfile("cn-shanghai", "您的AccessKey ID", "您的AccessKey Secret");
        DefaultProfile
                .addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        ListSimilarityImagesRequest request = new ListSimilarityImagesRequest();
        // 指定API返回格式、请求方法。
        request.setAcceptFormat(FormatType.JSON);
        request.setMethod(MethodType.POST);
        request.setEncoding("utf-8");
        request.setProtocol(ProtocolType.HTTP);
        // pageSize：每页大小，取值范围：(0,50]，currentPage：当前页码，取值范围：(0,50]。
        JSONObject httpBody = new JSONObject();
        httpBody.put("library", "相似图样本图库名称");
        httpBody.put("pageSize", "5");
        httpBody.put("currentPage", "1");
        request.setHttpContent(org.apache.commons.codec.binary.StringUtils.getBytesUtf8(htt
pBody.toJSONString()),
                "UTF-8", FormatType.JSON);
        /**
         * 请设置超时时间，服务端全链路处理超时时间为10秒，请做相应设置。
         * 如果您设置的ReadTimeout小于服务端处理的时间，程序中会获得一个ReadTimeout异常。
         */
        request.setConnectTimeout(3000);
        request.setReadTimeout(10000);
        HttpResponse httpResponse = null;
        try {
            httpResponse = client.doAction(request);
            // 服务端接收到请求，完成处理后返回的结果。
            if (httpResponse != null && httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(org.apache.commons.codec.binary.S
tringUtils.newStringUtf8(httpResponse.getHttpContent()));
                System.out.println(JSON.toJSONString(scrResponse, true));
                int requestCode = scrResponse.getIntValue("code");
                if (200 == requestCode) {
                    // 当前页码。
                    int currentPage = scrResponse.getIntValue("currentPage");
                    // 页面大小。
                    int pageSize = scrResponse.getIntValue("pageSize");
                    // 图库总数量。
                    int totalCount = scrResponse.getIntValue("totalCount");
                    // 具体结果列表。
                    JSONArray results = scrResponse.getJSONArray("data");
                    for (Object data : results) {
                        // 单个结果。
                        // 创建图片的时间戳。
                        int createTime = ((JSONObject) data).getIntValue("createTime");
                        // 图片的唯一ID。
```

```
                        String dataId = ((JSONObject) data).getString("dataId");
                        // 图片的链接地址。
                        String url = ((JSONObject) data).getString("url");
                        // 图片样本的标签。
                        String tags = ((JSONObject) data).getString("tags");
                    }
                } else {
                    /**
                     * 表明请求整体处理失败，原因视具体的情况详细分析。
                     */
                    System.out.println("the whole request failed. response:" + JSON.toJSONS
tring(scrResponse));
                }
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

2.7.1.7. 查询相似图样本图片详情

本文介绍了如何使用Java SDK查询相似图样本图片详情。

功能描述

调用该接口查询相似图样本图片详情。关于参数的详细说明，请参见查询样本图库详情API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Java依赖。关于安装Java依赖的具体操作，请参见安装Java依赖。

  > **说明** 请一定按照安装Java依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

查询相似图样本图片详情任务

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.green.model.v20180509.GetSimilarityImageRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.http.MethodType;
import com.aliyuncs.http.ProtocolType;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
public class GetSimilarityImageSample {
    public static void main(String[] args) throws Exception {
```

```
        IClientProfile profile = DefaultProfile
                .getProfile("cn-shanghai", "您的AccessKey ID", "您的AccessKey Secret");
        DefaultProfile
                .addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        GetSimilarityImageRequest request = new GetSimilarityImageRequest();
        // 指定API返回格式、请求方法。
        request.setAcceptFormat(FormatType.JSON);
        request.setMethod(MethodType.POST);
        request.setEncoding("utf-8");
        request.setProtocol(ProtocolType.HTTP);
        JSONObject httpBody = new JSONObject();
        httpBody.put("library", "相似图样本图库名称");
        httpBody.put("dataId", "样本图片ID");
        request.setHttpContent(org.apache.commons.codec.binary.StringUtils.getBytesUtf8(htt
pBody.toJSONString()),
                "UTF-8", FormatType.JSON);
        /**
         * 请设置超时时间，服务端全链路处理超时时间为10秒，请做相应设置。
         * 如果您设置的ReadTimeout小于服务端处理的时间，程序中会获得一个ReadTimeout异常。
         */
        request.setConnectTimeout(3000);
        request.setReadTimeout(10000);
        HttpResponse httpResponse = null;
        try {
            httpResponse = client.doAction(request);
            // 服务端接收到请求，完成处理后返回的结果。
            if (httpResponse != null && httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(org.apache.commons.codec.binary.S
tringUtils.newStringUtf8(httpResponse.getHttpContent()));
                System.out.println(JSON.toJSONString(scrResponse, true));
                int requestCode = scrResponse.getIntValue("code");
                JSONObject data = scrResponse.getJSONObject("data");
                if (200 == requestCode) {
                    // 创建图片的时间戳。
                    int createTime = data.getIntValue("createTime");
                    // 图片的唯一ID。
                    String dataId = data.getString("dataId");
                    // 图片的链接地址。
                    String url = data.getString("url");
                } else {
                    /**
                     * 表明请求整体处理失败，原因视具体的情况详细分析。
                     */
                    System.out.println("the whole request failed. response:" + JSON.toJSONS
tring(scrResponse));
                }
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

2.7.1.8. 移除相似图样本

本文介绍了如何使用Java SDK，将相似图样本从图库中移除。

功能描述

移除操作在1分钟之内生效。一次最多允许移除100张样本图片。关于参数的详细说明，请参见删除样本图片API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Java依赖。关于安装Java依赖的具体操作，请参见安装Java依赖。

  > **说明** 请一定按照安装Java依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

示例代码

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.green.model.v20180509.DeleteSimilarityImageRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.http.MethodType;
import com.aliyuncs.http.ProtocolType;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import java.util.*;
public class Main {
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile
            .getProfile("cn-shanghai", "您的Accesskey ID", "您的Accesskey Secret");
        DefaultProfile
            .addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        DeleteSimilarityImageRequest deleteSimilarityImageRequest = new DeleteSimilarityIma
geRequest();
        // 指定API返回格式、请求方法。
        deleteSimilarityImageRequest.setAcceptFormat(FormatType.JSON);
        deleteSimilarityImageRequest.setMethod(MethodType.POST);
        deleteSimilarityImageRequest.setEncoding("utf-8");
        deleteSimilarityImageRequest.setProtocol(ProtocolType.HTTP);
        JSONObject httpBody = new JSONObject();
        // 批量删除图库中的图片需要指定dataId，该接口是免费接口。
        httpBody.put("dataIds", Arrays.asList("x-3340-4b10-a622-7a5668f24b81"));
        deleteSimilarityImageRequest.setHttpContent(org.apache.commons.codec.binary.StringU
tils.getBytesUtf8(httpBody.toJSONString()),
```

```
            "UTF-8", FormatType.JSON);
        /**
         * 请设置超时时间，服务端全链路处理超时时间为10秒，请做相应设置。
         * 如果您设置的ReadTimeout小于服务端处理的时间，程序中会获得一个ReadTimeout异常。
         */
        deleteSimilarityImageRequest.setConnectTimeout(3000);
        deleteSimilarityImageRequest.setReadTimeout(10000);
        HttpResponse httpResponse = null;
        try {
            httpResponse = client.doAction(deleteSimilarityImageRequest);
        } catch (Exception e) {
            e.printStackTrace();
        }
        // 服务端接收到请求，并完成处理返回的结果。
        if (httpResponse != null && httpResponse.isSuccess()) {
            JSONObject scrResponse = JSON.parseObject(org.apache.commons.codec.binary.Strin
gUtils.newStringUtf8(httpResponse.getHttpContent()));
            System.out.println(JSON.toJSONString(scrResponse, true));
            int requestCode = scrResponse.getIntValue("code");
            // 每一张图片的检测结果。
            JSONArray taskResults = scrResponse.getJSONArray("data");
            if (200 == requestCode) {
                for (Object taskResult : taskResults) {
                    //单张图片的处理结果
                    int taskCode = ((JSONObject) taskResult).getIntValue("code");
                    if (200 == taskCode) {
                        System.out.println("成功删除1张图片 ");
                    } else {
                        // 单张图片处理失败，原因是具体的情况详细分析。
                        System.out.println("task process fail. task response:" + JSON.toJSO
NString(taskResult));
                    }
                }
            } else {
                /**
                 * 表明请求整体处理失败，原因视具体的情况详细分析。
                 */
                System.out.println("the whole image scan request failed. response:" + JSON.
toJSONString(scrResponse));
            }
        }
    }
}
```

2.7.1.9. 删除相似图库

本文介绍了如何使用Java SDK删除指定相似图库。

功能描述

只有当相似图库中无图片样本时，才允许被删除。关于参数的详细说明，请参见删除图库API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Java依赖。关于安装Java依赖的具体操作，请参见安装Java依赖。

  > **说明** 请一定按照安装Java依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

删除相似图库任务

```
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONArray;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.green.model.v20180509.DeleteSimilarityLibraryRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.http.MethodType;
import com.aliyuncs.http.ProtocolType;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
public class DeleteSimilarityLibrarySample {
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile
                .getProfile("cn-shanghai", "您的AccessKey ID", "您的AccessKey Secret");
        DefaultProfile
                .addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        DeleteSimilarityLibraryRequest request = new DeleteSimilarityLibraryRequest();
        // 指定API返回格式、请求方法。
        request.setAcceptFormat(FormatType.JSON);
        request.setMethod(MethodType.POST);
        request.setEncoding("utf-8");
        request.setProtocol(ProtocolType.HTTP);
        JSONObject httpBody = new JSONObject();
        httpBody.put("name", "相似图样本图库名称");
        request.setHttpContent(org.apache.commons.codec.binary.StringUtils.getBytesUtf8(htt
pBody.toJSONString()),
                "UTF-8", FormatType.JSON);
        /**
         * 请设置超时时间，服务端全链路处理超时时间为10秒，请做相应设置。
         * 如果您设置的ReadTimeout小于服务端处理的时间，程序中会获得一个ReadTimeout异常。
         */
        request.setConnectTimeout(3000);
        request.setReadTimeout(10000);
        HttpResponse httpResponse = null;
        try {
            httpResponse = client.doAction(request);
            // 服务端接收到请求，完成处理后返回的结果。
            if (httpResponse != null && httpResponse.isSuccess()) {
```

```
                JSONObject scrResponse = JSON.parseObject(org.apache.commons.codec.binary.S
tringUtils.newStringUtf8(httpResponse.getHttpContent()));
                System.out.println(JSON.toJSONString(scrResponse, true));
                int requestCode = scrResponse.getIntValue("code");
                if (200 == requestCode) {
                    // 删除成功。
                } else {
                    /**
                     * 表明请求整体处理失败，原因视具体的情况详细分析。
                     */
                    System.out.println("the whole scan request failed. response:" + JSON.to
JSONString(scrResponse));
                }
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

2.7.2. 自定义文本库

本文介绍了如何使用Java SDK管理自定义文本库，以满足文本反垃圾检测场景的个性化需求。

功能描述

根据文本类型的不同，文本库分为关键词文本库和相似文本库；根据管控目的不同，文本库分为白名单、黑名单、疑似名单。关于参数的详细信息，请参见自定义文本库API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Java依赖。关于安装Java依赖的具体操作，请参见安装Java依赖。

  > **说明** 请一定按照安装Java依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

获取文本库列表

- 获取关键词文本库列表

```
DescribeKeywordLibRequest describeKeywordLibRequest = new DescribeKeywordLibRequest();
        describeKeywordLibRequest.setServiceModule("open_api");
        try {
            // 将返回所有文本库，包括文本反垃圾的关键词文本库、文本反垃圾的相似文本库、图片广告的关键词文本库、语音反垃圾的关键词文本库。
            DescribeKeywordLibResponse describeKeywordLibResponse = client.getAcsResponse(describeKeywordLibRequest);
            System.out.println(JSON.toJSONString(describeKeywordLibResponse));
            // 过滤出文本反垃圾场景配置的文本库。
            List<DescribeKeywordLibResponse.KeywordLib> allLibs = describeKeywordLibResponse.getKeywordLibList();
            List<DescribeKeywordLibResponse.KeywordLib> textAntispamKeywordLibs = new ArrayList<DescribeKeywordLibResponse.KeywordLib>();
            for (DescribeKeywordLibResponse.KeywordLib keywordLib : allLibs) {
                String LibType = keywordLib.getLibType();
                String resourceType = keywordLib.getResourceType();
                String source =  keywordLib.getSource();
                // 获取文本反垃圾自定义的关键词文本库。
                if ("textKeyword".equals(LibType) && "TEXT".equals(resourceType)  && "MANUAL".equals(source)) {
                    textAntispamKeywordLibs.add(keywordLib);
                }
                // 获取文本反垃圾回流的关键词文本库。
                if ("textKeyword".equals(LibType) && "TEXT".equals(resourceType)  && "FEEDBACK".equals(source)) {
                    textAntispamKeywordLibs.add(keywordLib);
                }
            }
            System.out.println(JSON.toJSONString(textAntispamKeywordLibs));
        } catch (ClientException e) {
            e.printStackTrace();
        }
```

- 获取相似文本库列表（包含自定义和系统回流的相似文本库）

```
DescribeKeywordLibRequest describeKeywordLibRequest = new DescribeKeywordLibRequest();
        describeKeywordLibRequest.setServiceModule("open_api");
        try {
            // 将返回所有文本库，包括文本反垃圾的关键词文本库、文本反垃圾的相似文本库、图片广告的关键词文本库、语音反垃圾的关键词文本库。
            DescribeKeywordLibResponse describeKeywordLibResponse = client.getAcsResponse(describeKeywordLibRequest);
            System.out.println(JSON.toJSONString(describeKeywordLibResponse));
            // 过滤出相似文本库。
            List<DescribeKeywordLibResponse.KeywordLib> allLibs = describeKeywordLibResponse.getKeywordLibList();
            List<DescribeKeywordLibResponse.KeywordLib> similarTextLibs = new ArrayList<DescribeKeywordLibResponse.KeywordLib>();
            for (DescribeKeywordLibResponse.KeywordLib keywordLib : allLibs) {
                String LibType =  keywordLib.getLibType();
                String resourceType = keywordLib.getResourceType();
                String source =  keywordLib.getSource();
                // 获取文本反垃圾自定义的相似文本库。
                if("similarText".equals(LibType) && "TEXT".equals(resourceType)  && "MANUAL".equals(source)) {
                    similarTextLibs.add(keywordLib);
                }
                // 获取文本反垃圾回流的相似文本库。
                if("similarText".equals(LibType) && "TEXT".equals(resourceType)  && "FEEDBACK".equals(source)) {
                    similarTextLibs.add(keywordLib);
                }
            }
            System.out.println(JSON.toJSONString(similarTextLibs));
        } catch (ClientException e) {
            e.printStackTrace();
        }
```

创建文本库

- 创建关键词文本库

```
CreateKeywordLibRequest createKeywordLibRequest = new CreateKeywordLibRequest();
        createKeywordLibRequest.setServiceModule("open_api");
        createKeywordLibRequest.setName("测试关键词文本库");
        // 设置为文本反垃圾场景使用。
        createKeywordLibRequest.setResourceType("TEXT");
        // 设置类型为关键词。
        createKeywordLibRequest.setLibType("textKeyword");
        // 设置创建黑库。
        createKeywordLibRequest.setCategory("BLACK");
        try {
            CreateKeywordLibResponse describeKeywordLibResponse = client.getAcsResponse(c
reateKeywordLibRequest);
            // 关键词文本库ID。无异常则表示创建成功。
            String libId = describeKeywordLibResponse.getId();
        } catch (ClientException e) {
            e.printStackTrace();
        }
```

- 创建相似文本库

```
CreateKeywordLibRequest createKeywordLibRequest = new CreateKeywordLibRequest();
        createKeywordLibRequest.setServiceModule("open_api");
        createKeywordLibRequest.setName("测试相似文本库");
        // 设置为文本反垃圾场景使用。
        createKeywordLibRequest.setResourceType("TEXT");
        // 设置类型为相似文本。
        createKeywordLibRequest.setLibType("similarText");
        // 相似文本库支持创建黑名单库、白名单、疑似名单。
        createKeywordLibRequest.setCategory("BLACK");
        try {
            CreateKeywordLibResponse describeKeywordLibResponse = client.getAcsResponse(c
reateKeywordLibRequest);
            // 相似文本库ID。无异常则表示创建成功。
            String libId = describeKeywordLibResponse.getId();
        } catch (ClientException e) {
            e.printStackTrace();
        }
```

修改文本库

更新文本库库名以及修改文本库所使用的检测场景。

```
UpdateKeywordLibRequest updateKeywordLibRequest = new UpdateKeywordLibRequest();
        // 设置待操作的文本库ID。
        updateKeywordLibRequest.setId(0);
        // 设置新的文本库名称。
        updateKeywordLibRequest.setName("修改后的名称");
        // 设置新的BizType。
        updateKeywordLibRequest.setBizTypes(JSON.toJSONString(Arrays.asList("新的BizType_1",
"新的BizType_2")));
        try {
            UpdateKeywordLibResponse updateKeywordLibResponse = client.getAcsResponse(updat
eKeywordLibRequest);
            // 请求ID。无异常则表示修改成功。
            String requestId = updateKeywordLibResponse.getRequestId();
        } catch (ClientException e) {
            e.printStackTrace();
        }
```

删除文本库

> **注意** 删除文本库也将删除文本库下的文本。系统回流的文本库不允许删除。

```
DeleteKeywordLibRequest deleteKeywordLibRequest = new DeleteKeywordLibRequest();
        // 设置待删除的文本库ID。
        deleteKeywordLibRequest.setId(3353);
        try {
            DeleteKeywordLibResponse deleteKeywordLibResponse = client.getAcsResponse(delet
eKeywordLibRequest);
            // 请求ID。无异常则表示删除成功。
            String requestId = deleteKeywordLibResponse.getRequestId();
        } catch (ClientException e) {
            e.printStackTrace();
        }
```

查找文本

默认分页获取所有文本。如果设置了Keyword字段，将模糊查找包含该字段值的文本。

```
DescribeKeywordRequest describeKeywordRequest = new DescribeKeywordRequest();
        // 设置待查询的文本库ID。
        describeKeywordRequest.setKeywordLibId(0);
        describeKeywordRequest.setPageSize(10);
        describeKeywordRequest.setCurrentPage(1);
        // 可选，用于模糊查。
        describeKeywordRequest.setKeyword("查询文本");
        try {
            DescribeKeywordResponse describeKeywordResponse = client.getAcsResponse(describ
eKeywordRequest);
            // 总数。
            Integer totalCount = describeKeywordResponse.getTotalCount();
            // 当前页。
            Integer currentPage = describeKeywordResponse.getCurrentPage();
            // 分页大小。
            Integer pageSize = describeKeywordResponse.getPageSize();
            for (DescribeKeywordResponse.Keyword keywordItem : describeKeywordResponse.getK
eywordList()) {
                // 关键词主键ID。
                Integer id = keywordItem.getId();
                // 创建时间。
                String createTime = keywordItem.getCreateTime();
                // 关键词的匹配命中次数。
                Integer hitCount = keywordItem.getHitCount();
                // 关键词。
                String keyword = keywordItem.getKeyword();
            }
        } catch (ClientException e) {
            e.printStackTrace();
        }
```

添加文本

```
CreateKeywordRequest createKeywordRequest = new CreateKeywordRequest();
        // 设置待操作的文本库ID。
        createKeywordRequest.setKeywordLibId(0L);
        // 待添加的文本。
        createKeywordRequest.setKeywords(JSON.toJSONString(Arrays.asList("关键词_1", "关键词_
2")));
        try {
            CreateKeywordResponse createKeywordResponse = client.getAcsResponse(createKeywo
rdRequest);
            // 已成功添加的关键词数量。
            Integer successCount = createKeywordResponse.getSuccessCount();
            // 无效的关键词列表，即未添加成功的关键词。
            List<String> invalidKeywordList = createKeywordResponse.getInvalidKeywordList()
;
        } catch (ClientException e) {
            e.printStackTrace();
        }
```

删除文本

```
DeleteKeywordRequest deleteKeywordRequest = new DeleteKeywordRequest();
        // 设置待操作的文本库ID。
        deleteKeywordRequest.setKeywordLibId(String.valueOf("文本库ID"));
        // 待删除的文本。
        deleteKeywordRequest.setIds(JSON.toJSONString(Arrays.asList(1, 2)));
        try {
            DeleteKeywordResponse deleteKeywordResponse = client.getAcsResponse(deleteKeywo
rdRequest);
            // 请求ID。无异常则表示删除成功。
            String requestId = deleteKeywordResponse.getRequestId();
        } catch (ClientException e) {
            e.printStackTrace();
        }
```

2.7.3. 自定义图库

本文介绍了如何使用Java SDK管理自定义图库。

功能描述

您可以自定义智能鉴黄、暴恐涉政识别、图片或视频广告的图片样本，满足个性化内容管控需求。关于参数的详细信息，请参见创建图库API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Java依赖。关于安装Java依赖的具体操作，请参见安装Java依赖。

  > **说明** 请一定按照安装Java依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

获取自定义图库列表

您可以使用以下代码获取用户图库列表（包括用户自定义的图库列表和系统回流图库）：

```
DescribeImageLibRequest describeImageLibRequest = new DescribeImageLibRequest();
describeImageLibRequest.setServiceModule("open_api");
try {
     // 获取所有的图库，包括自定义的图库以及回流图库。
     DescribeImageLibResponse describeImageLibResponse = client.getAcsResponse(describeImag
eLibRequest);
     System.out.println(JSON.toJSONString(describeImageLibResponse));
     List<DescribeImageLibResponse.ImageLib> allLibs = describeImageLibResponse.getImageLib
List();
     List<DescribeImageLibResponse.ImageLib> customImageLibs = new ArrayList<DescribeImageL
ibResponse.ImageLib>();
     for (DescribeImageLibResponse.ImageLib imageLib : allLibs) {
     String source =  imageLib.getSource();
     // 过滤出用户自定义的图库。
     if("MANUAL".equals(source)) {
         customImageLibs.add(imageLib);
        }
      // 过滤出系统回流（反馈）的图库。
      if("FEEDBACK".equals(source)) {
          customImageLibs.add(imageLib);
        }
      }
     System.out.println(JSON.toJSONString(customImageLibs));
} catch (ClientException e) {
            e.printStackTrace();
}
```

创建自定义图库

您可以使用以下代码创建自定义图库：

> **说明** 请根据您所需要的业务场景设置不同的参数。

```
CreateImageLibRequest  createImageLibRequest = new CreateImageLibRequest();
createImageLibRequest.setServiceModule("open_api");
createImageLibRequest.setName("鉴黄黑库");
createImageLibRequest.setScene("PORN");
createImageLibRequest.setCategory("BLACK");
try {
    CreateImageLibResponse createImageLibResponse = client.getAcsResponse(createImageLibReq
uest);
    // 请求ID，无异常则表示创建成功。
    String requestId = createImageLibResponse.getRequestId();
    System.out.println(JSON.toJSONString(createImageLibResponse));
  } catch (ClientException e) {
      e.printStackTrace();
  }
```

修改自定义图库

您可以使用以下代码修改自定义图库的名称及其适用的业务场景（BizType）：

```
UpdateImageLibRequest updateImageLibRequest = new UpdateImageLibRequest();
// 图库ID。
updateImageLibRequest.setId(12345);
updateImageLibRequest.setName("鉴黄黑库改名");
updateImageLibRequest.setBizTypes(JSON.toJSONString(Arrays.asList("comment")));
updateImageLibRequest.setCategory("WHITE");
updateImageLibRequest.setScene("PORN");
try {
    UpdateImageLibResponse updateImageLibResponse = client.getAcsResponse(updateImageLibReq
uest);
    // 请求ID，无异常则表示修改成功。
    String requestId = updateImageLibResponse.getRequestId();
    System.out.println(JSON.toJSONString(updateImageLibResponse));
} catch (ClientException e) {
    e.printStackTrace();
}
```

删除自定义图库

您可以使用以下代码删除自定义图库：

> **说明** 删除自定义图库时，图库下的所有图片也将被删除。

```
DeleteImageLibRequest  deleteImageLibRequest = new DeleteImageLibRequest();
// 图库ID。
deleteImageLibRequest.setId(12345);
try {
     DeleteImageLibResponse deleteImageLibResponse = client.getAcsResponse(deleteImageLibRe
quest);
     // 请求ID，无异常则表示删除成功。
     String requestId = deleteImageLibResponse.getRequestId();
     System.out.println(JSON.toJSONString(deleteImageLibResponse));
} catch (ClientException e) {
    e.printStackTrace();
}
```

获取自定义图库图片列表

您可以使用以下代码获取自定义图库中所有已添加的图片列表：

```
DescribeImageFromLibRequest describeImageFromLibRequest = new DescribeImageFromLibRequest()
;
describeImageFromLibRequest.setImageLibId(1519);
describeImageFromLibRequest.setPageSize(20);
describeImageFromLibRequest.setCurrentPage(1);
try {
    DescribeImageFromLibResponse describeImageFromLibResponse = client.getAcsResponse(descr
ibeImageFromLibRequest);
    // 返回图片数
    describeImageFromLibResponse.getTotalCount();
    // 返回的当前分页
    describeImageFromLibResponse.getCurrentPage();
    // 返回的分页大小，即每个分页显示的图片数量
    describeImageFromLibResponse.getPageSize();
    // 图片列表
    for (DescribeImageFromLibResponse.ImageFromLib imageFromLib : describeImageFromLibRespo
nse.getImageFromLibList()) {
          // 图片主键ID
          imageFromLib.getId();
          // 图片URL地址
          imageFromLib.getImage();
          // 图片的缩略图URL地址
          imageFromLib.getThumbnail();
     }
     System.out.println(JSON.toJSONString(describeImageFromLibResponse));
} catch (ClientException e) {
    e.printStackTrace();
}
```

向自定义图库中添加图片

您可以使用以下代码向自定义图库中添加图片。添加图片的过程包括：

1. 获取图片上传凭证。
2. 上传图片到图片空间。
3. 提交上传后的库信息、图片路径信息到服务器。

```
 // 获取上传凭证。
DescribeUploadInfoRequest describeUploadInfoRequest = new DescribeUploadInfoRequest();
describeUploadInfoRequest.setBiz("customImageLib");
DescribeUploadInfoResponse describeUploadInfoResponse = null;
try {
     describeUploadInfoResponse = client.getAcsResponse(describeUploadInfoRequest);
     System.out.println(JSON.toJSONString(describeUploadInfoResponse));
} catch (ClientException e) {
            e.printStackTrace();
}
// 上传图片。
CustomLibUploader customLibUploader = new CustomLibUploader();
String object = null;
try {
    object = customLibUploader.uploadFile(describeUploadInfoResponse.getHost(), describeUpl
oadInfoResponse.getFolder(), describeUploadInfoResponse.getAccessid(),
             describeUploadInfoResponse.getPolicy(), describeUploadInfoResponse.getSignatur
e(),
             "/Users/liuhai.lh/Desktop/a.jpg");
} catch (Exception e) {
    e.printStackTrace();
}
if(org.apache.commons.lang.StringUtils.isNotBlank(object)){
    UploadImageToLibRequest imageToLibRequest = new UploadImageToLibRequest();
    imageToLibRequest.setImageLibId(1519);
    imageToLibRequest.setImages(JSON.toJSONString(Arrays.asList(object)));
    try {
        UploadImageToLibResponse uploadImageToLibResponse = client.getAcsResponse(imageToLi
bRequest);
        // 请求ID，无异常则表示添加成功。
        String requestId = uploadImageToLibResponse.getRequestId();
        System.out.println(JSON.toJSONString(uploadImageToLibResponse));
    } catch (ClientException e) {
        e.printStackTrace();
    }
}
```

删除自定义图片

您可以使用以下代码删除自定义图库中的多张自定义图片：

```
DeleteImageFromLibRequest deleteImageFromLibRequest = new DeleteImageFromLibRequest();
deleteImageFromLibRequest.setIds(JSON.toJSONString(Arrays.asList(669310)));
try {
    DeleteImageFromLibResponse deleteImageFromLibResponse = client.getAcsResponse(deleteIma
geFromLibRequest);
    // 请求ID，无异常则表示删除成功。
    String requestId = uploadImageToLibResponse.getRequestId();
    System.out.println(JSON.toJSONString(deleteImageFromLibResponse));
} catch (ClientException e) {
    e.printStackTrace();
}
```

2.7.4. OSS内容检测

本文介绍了如何使用Java SDK检测OSS违规的内容。

前提条件

- 安装Java依赖。关于安装Java依赖的具体操作，请参见安装Java依赖。

  > **说明** 请一定按照安装Java依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

获取OSS违规检测数据

您可以参考以下代码示例获取OSS违规检测数据：

> **说明** 以下代码仅为简单示例，具体的接口参数请参考OSS违规检测API。

```
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.green.model.v20170823.DescribeOssResultItemsRequest;
import com.aliyuncs.green.model.v20170823.DescribeOssResultItemsResponse;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
public class Main {
    public static void main(String[] args) {
        // 请替换成您的AccessKey ID、AccessKey Secret。
        IClientProfile profile = DefaultProfile
                .getProfile("cn-shanghai", "您的AccessKey ID", "您的AccessKey Secret");
        DefaultProfile
                .addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        DescribeOssResultItemsRequest describeKeywordLibRequest = new DescribeOssResultItem
sRequest();
        describeKeywordLibRequest.setResourceType("VIDEO");
        describeKeywordLibRequest.setScene("porn");
        describeKeywordLibRequest.setStock(false);
        describeKeywordLibRequest.setStartDate("2018-12-07 00:00:00 +0800");
        describeKeywordLibRequest.setEndDate("2018-12-13 15:00:53 +0800");
        try {
            DescribeOssResultItemsResponse describeOssResultItemsResponse = client.getAcsRe
sponse(describeKeywordLibRequest);
            // 结果总数。
            describeOssResultItemsResponse.getTotalCount();
            // 当前页码。
            describeOssResultItemsResponse.getCurrentPage();
            // 分页大小。
            describeOssResultItemsResponse.getPageSize();
            // 扫描结果列表。
            for (DescribeOssResultItemsResponse.ScanResult scanResult : describeOssResultIt
emsResponse.getScanResultList()) {
                // 资源所在的OSS Bucket。
```

```
                scanResult.getBucket();
                // 数据ID。
                scanResult.getDataId();
                // 资源的处理状态
                scanResult.getHandleStatus();
                // 扫描结果的唯一标识。
                scanResult.getId();
                // 原始文件链接地址。
                scanResult.getNewUrl();
                // OSS Object名称。
                scanResult.getObject();
                // 请求时间。
                scanResult.getRequestTime();
                // 资源的状态
                scanResult.getResourceStatus();
                // 扫描完成时间。
                scanResult.getScanFinishedTime();
                scanResult.getScore();
                // 建议您执行的后续操作。
                scanResult.getSuggestion();
                // 任务ID。
                scanResult.getTaskId();
                // 缩略图链接。
                scanResult.getThumbnail();
                // （仅使用视频对象）存在风险的问题帧。
                for (DescribeOssResultItemsResponse.ScanResult.FrameResult frameResult : sc
anResult.getFrameResults()) {
                    // 该帧的位置，即距离视频开头的偏移量，单位为秒。
                    frameResult.getOffset();
                    // 该帧的缩略图链接。
                    frameResult.getThumbnail();
                    // 该帧的链接。
                    frameResult.getUrl();
                }
            }
        } catch (ClientException e) {
            e.printStackTrace();
        }
    }
}
```

对OSS的审核结果进行标记

该接口能够对OSS的扫描结果进行标记和操作。如果需要对已检测出结果的内容执行删除、标记为正常并忽略，或者解除冻结等操作，您可以调用本接口。

> **说明** 以下代码仅为简单示例，具体的接口参数请参考OSS违规检测API。

```
import com.alibaba.fastjson.JSON;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.green.model.v20170823.MarkOssResultRequest;
import com.aliyuncs.green.model.v20170823.MarkOssResultResponse;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import java.util.Arrays;
public class Main {
    public static void main(String[] args) {
        // 请替换成您的AccessKey ID、AccessKey Secret。
        IClientProfile profile = DefaultProfile
                .getProfile("cn-shanghai", "您的AccessKey ID", "您的AccessKey Secret");
        DefaultProfile
                .addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        MarkOssResultRequest markOssResultRequest = new MarkOssResultRequest();
        markOssResultRequest.setResourceType("VIDEO");
        markOssResultRequest.setScene("terrorism ");
        markOssResultRequest.setStock(false);
        markOssResultRequest.setIds(JSON.toJSONString(Arrays.asList(24930001L)));
        markOssResultRequest.setOperation("ignore");
        try {
            MarkOssResultResponse markOssResultResponse = client.getAcsResponse(markOssResu
ltRequest);
            // 请求ID，无异常则表示创建成功。
            String requestId = markOssResultResponse.getRequestId();
            System.out.println("mark success." + JSON.toJSONString(markOssResultResponse));
        } catch (ClientException e) {
            e.printStackTrace();
        }
    }
}
```

以文件形式导出OSS违规检测结果

您可以参考以下代码示例通过文件形式导出OSS违规检测结果：

> **说明** 以下代码仅为简单示例，具体的接口参数请参考OSS违规检测API。

```
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.green.model.v20170823.ExportOssResultRequest;
import com.aliyuncs.green.model.v20170823.ExportOssResultResponse;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
public class Main {
    public static void main(String[] args) {
        // 请替换成您的AccessKey ID、AccessKey Secret。
        IClientProfile profile = DefaultProfile
                .getProfile("cn-shanghai", "您的AccessKey ID", "您的AccessKey Secret");
        DefaultProfile
                .addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        ExportOssResultRequest exportOssResultRequest = new ExportOssResultRequest();
        exportOssResultRequest.setResourceType("VIDEO");
        exportOssResultRequest.setScene("porn");
        exportOssResultRequest.setStock(false);
        exportOssResultRequest.setStartDate("2018-12-07 00:00:00 +0800");
        exportOssResultRequest.setEndDate("2018-12-13 15:00:53 +0800");
        try {
            ExportOssResultResponse exportOssResultResponse = client.getAcsResponse(exportO
ssResultRequest);
            // 返回文件的链接地址。
            String fileUrl = exportOssResultResponse.getFileUrl();
            System.out.println("export success. file url:" + fileUrl);
        } catch (ClientException e) {
            e.printStackTrace();
        }
    }
}
```

2.7.5. 业务场景管理

2.7.5.1. 创建业务场景

本文介绍了如何通过Java SDK创建业务场景，业务场景主要用于内容检测API功能自定义机审标准。

前提条件

- 安装Java依赖。关于安装Java依赖的具体操作，请参见安装Java依赖。

  > **说明** 请一定按照安装Java依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

创建业务场景

接口	描述	支持的地域
CreateBizType	创建业务场景。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
// 引入创建业务场景需要的类。
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.green.model.v20170823.CreateBizTypeRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
/**
 * 通过如下代码创建业务场景。
 */
public class CreateBizTypeRequestSample {
    public static void main(String[] args) throws Exception {
        // 请替换成您的AccessKey ID、AccessKey Secret。
        IClientProfile profile = DefaultProfile
                .getProfile("cn-shanghai", "您的AccessKey ID", "您的AccessKey Secret");
        DefaultProfile
                .addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        CreateBizTypeRequest createBizTypeRequest = new CreateBizTypeRequest();
        // 创建业务场景的名称。
        createBizTypeRequest.setBizTypeName("您要创建的业务场景");
        // 导入已经创建的业务场景对应的配置。可以不设置。
        createBizTypeRequest.setBizTypeImport("您已经创建的业务场景");
        // 是否引入行业模板配置。当值为true时必须传industryInfo。可以不设置。
        createBizTypeRequest.setCiteTemplate(true);
        // 行业分类。CiteTemplate为true时，必须设置。取值范围：社交-注册信息-头像；社交-注册信息-昵
称。
        createBizTypeRequest.setIndustryInfo("社交-注册信息-头像");
        // 指定API返回格式。
        createBizTypeRequest.setAcceptFormat(FormatType.JSON);
        // 指定请求方法。
        createBizTypeRequest.setMethod(com.aliyuncs.http.MethodType.POST);
        createBizTypeRequest.setEncoding("utf-8");
        // 请务必设置连接超时时间，您可根据实际情况进行修改。单位：ms。
        createBizTypeRequest.setConnectTimeout(3000);
        // 请务必设置超时时间，您可根据实际情况进行修改。单位：ms。
        createBizTypeRequest.setReadTimeout(6000);
        // HTTP调用创建业务场景。
```

```
        try {
            HttpResponse httpResponse = client.doAction(createBizTypeRequest);
            // 判断HTTP请求是否成功。
            if (httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttpCo
ntent(), "utf-8"));
                System.out.println("create success." + JSON.toJSONString(scrResponse, true)
);
            } else {
                JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttpCo
ntent(), "utf-8"));
                // 失败原因。
                System.out.println("create not success. " + JSON.toJSONString(scrResponse.g
et("Message"), true));
            }
        } catch (ServerException e) {
            e.printStackTrace();
        } catch (ClientException e) {
            e.printStackTrace();
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

2.7.5.2. 删除业务场景

本文介绍了如何通过Java SDK删除业务场景。如果您需要删除业务场景，请务必保证业务场景下无数据调用再删除，否则删除业务场景后对应的策略配置恢复为默认，有可能导致业务数据误识别。

前提条件

- 安装Java依赖。关于安装Java依赖的具体操作，请参见安装Java依赖。

  > **说明** 请一定按照安装Java依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

删除业务场景

接口	描述	支持的地域
DeleteBizType	删除业务场景。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
// 引入需要的类。
import com.alibaba.fastjson.JSON;
```

```
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.green.model.v20170823.DeleteBizTypeRequest;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
/**
 * 通过如下代码删除业务场景。
 */
public class DeleteBizTypeRequestSample {
    public static void main(String[] args) throws Exception {
        // 请替换成您的AccessKey ID、AccessKey Secret。
        IClientProfile profile = DefaultProfile
                .getProfile("cn-shanghai", "您的AccessKey ID", "您的AccessKey Secret");
        DefaultProfile
                .addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        DeleteBizTypeRequest deleteBizTypeRequest = new DeleteBizTypeRequest();
        // 设置需要删除业务场景名称。
        deleteBizTypeRequest.setBizTypeName("您要删除的业务场景");
        // 指定API返回格式。
        deleteBizTypeRequest.setAcceptFormat(FormatType.JSON);
        // 指定请求方法。
        deleteBizTypeRequest.setMethod(com.aliyuncs.http.MethodType.POST);
        deleteBizTypeRequest.setEncoding("utf-8");
        // 请务必设置连接超时时间，您可根据实际情况进行修改。单位：ms。
        deleteBizTypeRequest.setConnectTimeout(3000);
        // 请务必设置超时时间，您可根据实际情况进行修改。单位：ms。
        deleteBizTypeRequest.setReadTimeout(6000);
        // HTTP调用删除业务场景。
        try {
            HttpResponse httpResponse = client.doAction(deleteBizTypeRequest);
            // 判断HTTP请求是否成功。
            if (httpResponse.isSuccess()) {
                JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttpCo
ntent(), "utf-8"));
                System.out.println("delete success. detail msg:" + JSON.toJSONString(scrRes
ponse, true));
            } else {
                JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttpCo
ntent(), "utf-8"));
                // 失败原因。
                System.out.println("delete fail. detail msg: " + JSON.toJSONString(scrRespo
nse.get("Message"), true));
            }
        } catch (ServerException e) {
            e.printStackTrace();
        } catch (ClientException e) {
            e.printStackTrace();
        } catch (Exception e) {
```

```
            e.printStackTrace();
        }
    }
}
```

2.7.5.3. 查询业务场景

本文介绍了如何通过Java SDK查询已创建和自定义的业务场景列表，用于在后台管理业务场景数据。

前提条件

- 安装Java依赖。关于安装Java依赖的具体操作，请参见安装Java依赖。

  > **说明** 请一定按照安装Java依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

查询业务场景

接口	描述	支持的地域
DescribeUserBizTypes	查询业务场景。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
// 引入需要的类。
import com.alibaba.fastjson.JSON;
import com.alibaba.fastjson.JSONObject;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ClientException;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.green.model.v20170823.DescribeUserBizTypesRequest;
import com.aliyuncs.green.model.v20170823.DescribeUserBizTypesResponse;
import com.aliyuncs.green.model.v20170823.DescribeUserBizTypesResponse.Item;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.http.HttpResponse;
import com.aliyuncs.http.MethodType;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import org.apache.commons.collections.CollectionUtils;
/**
 * 通过如下代码查询业务场景。
 */
public class DescribeBizTypesRequestSample {
    public static void main(String[] args) throws Exception {
        // 请替换成您的AccessKey ID、AccessKey Secret。
        IClientProfile profile = DefaultProfile
```

```
            .getProfile("cn-shanghai", "您的AccessKey ID", "您的AccessKey Secret");
        DefaultProfile
            .addEndpoint("cn-shanghai", "Green", "green.cn-shanghai.aliyuncs.com");
        IAcsClient client = new DefaultAcsClient(profile);
        DescribeUserBizTypesRequest describeUserBizTypesRequest = new DescribeUserBizTypesR
equest();
        // 指定API返回格式。
        describeUserBizTypesRequest.setAcceptFormat(FormatType.JSON);
        // 指定请求方法。
        describeUserBizTypesRequest.setMethod(MethodType.GET);
        describeUserBizTypesRequest.setEncoding("utf-8");
        // 请务必设置连接超时时间，您可根据实际情况进行修改。单位：ms。
        describeUserBizTypesRequest.setConnectTimeout(3000);
        // 请务必设置超时时间，您可根据实际情况进行修改。单位：ms。
        describeUserBizTypesRequest.setReadTimeout(6000);
        try {
            HttpResponse httpResponse = client.doAction(describeUserBizTypesRequest);
            // 判断HTTP请求是否成功。
            if (httpResponse.isSuccess()) {
                DescribeUserBizTypesResponse describeUserBizTypesResponse = JSON.parseObjec
t(new String(httpResponse.getHttpContent(), "UTF-8"), DescribeUserBizTypesResponse.class);
                // 所有业务场景列表。
                System.out.println("query success. bizTypes size :" + describeUserBizTypesR
esponse.getBizTypeList().size());
                if (CollectionUtils.isNotEmpty(describeUserBizTypesResponse.getBizTypeList(
))) {
                    for (Item bizTypeItem: describeUserBizTypesResponse.getBizTypeList()) {
                        // 业务场景名称。
                        System.out.println(bizTypeItem.getBizType());
                        // 是否属于引用行业模板。
                        System.out.println(bizTypeItem.getCiteTemplate());
                        // 行业信息。
                        System.out.println(bizTypeItem.getIndustryInfo());
                        // 业务场景来源。客户自定义：custom；内容安全服务默认配置：system。
                        System.out.println(bizTypeItem.getSource());
                    }
                }
                // 新建业务场景时，可导入业务场景列表。
                System.out.println("query success. import bizTypes size :" + describeUserBi
zTypesResponse.getBizTypeList().size());
            } else {
                JSONObject scrResponse = JSON.parseObject(new String(httpResponse.getHttpCo
ntent(), "utf-8"));
                // 失败原因。
                System.out.println("query fail. detail msg: " + JSON.toJSONString(scrRespon
se.get("Message"), true));
            }
        } catch (ServerException e) {
            e.printStackTrace();
        } catch (ClientException e) {
            e.printStackTrace();
        } catch (Exception e) {
            e.printStackTrace();
        }
```

```
        }
    }
```

3.Python SDK

3.1. 安装

在使用内容检测API Python SDK前，您需要通过pip安装相关的依赖。

前提条件

- 使用Python 2.x、Python 3.x。
- 确保源服务器已安装Python PIP依赖库。

  以安装python3-pip依赖库为例，Linux部分发行版的安装命令如下。

  - CentOS、Red Hat Enterprise Linux：

    ```
    yum -y install python3-pip
    ```

  - Ubuntu、Debian：

    ```
    apt-get -y install python3-pip
    ```

  - OpenSUSE、SUSE：

    ```
    zypper -n install python3-pip
    ```

安装SDK

- 如果您使用Python 2.x，执行以下命令，安装阿里云SDK核心库：

  ```
  sudo pip install aliyun-python-sdk-core==2.13.10
  pip install -v aliyun-python-sdk-green==3.6.5 //安装指定的版本。
  pip install oss2 //安装OSS依赖包。
  ```

- 如果您使用Python 3.x，执行以下命令，安装阿里云SDK核心库：

  ```
  sudo pip install aliyun-python-sdk-core-v3==2.13.10
  pip install -v aliyun-python-sdk-green==3.6.5 //安装指定的版本。
  pip install oss2 //安装OSS依赖包。
  ```

- 如果您需要检测本地文件或者二进制文件，请下载并在项目中引入Extension.Uploader工具类。

3.2. 初始化

您需要创建一个Client实例去发起API请求，并根据需要修改Client的配置参数。

背景信息

新建Client时，需要指定服务地域（Region）和阿里云账号的AccessKey信息（包括AccessKey ID和AccessKey Secret）。

- 关于服务地域（Region）的更多信息，请参见接入区域。
- 关于阿里云账号的AccessKey的更多信息，请参见获取AccessKey。

图片、语音、视频、文本、视频识别的Client

支持的Region包括：

- cn-shanghai
- cn-beijing
- ap-southeast-1

您可以使用以下代码创建Client，用于图片、语音、视频、文本检测：

```
from aliyunsdkcore import client
# 请替换成您的AccessKey Id、AccessKey Secret，您可以将其配置在配置文件里面，也可以直接明文替换
clt = client.AcsClient("您的AccessKey Id", "您的AccessKey Secret","cn-shanghai")
```

自定义图片库、关键词词库、相似文本库的Client

支持的region包括：cn-shanghai。通过cn-shanghai 区域添加即可，其他区域服务端会自动同步数据。

您可以使用以下代码创建Client，用于管理自定义图片库、自定义关键词词库、自定义相似文本库：

```
from aliyunsdkcore import client
# 请替换成您的accessKeyId、accessKeySecret，您可以将其配置在配置文件里面，也可以直接明文替换
clt = client.AcsClient("您的AccessKey Id", "您的AccessKey Secret","cn-shanghai")
```

3.3. 内容审核（机审）

3.3.1. 图片审核

本文介绍了如何使用Python SDK图片审核接口，检测图片中是否包含风险内容。

功能描述

图片审核支持同步检测和异步检测两种方式。

- 同步检测实时返回检测结果。关于参数的详细信息，请参见同步检测。
- 异步检测需要您轮询结果或者通过callback回调通知获取检测结果。关于参数的详细信息，请参见异步检测。

前提条件

- 安装Python依赖。关于安装Python依赖的具体操作，请参见安装Python依赖。

  > **说明** 请一定按照安装Python依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

（推荐）同步图片检测

接口	描述	支持的地域
ImageSyncScanRequest	提交图片同步检测任务，对图片进行多个风险场景的识别，包括色情、暴恐涉政、广告、二维码、不良场景、Logo（商标台标）识别。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

- 传图片URL进行检测

```
#coding=utf-8
# 以下代码用于调用图片同步检测接口并实时返回检测结果。
from aliyunsdkcore import client
from aliyunsdkgreen.request.v20180509 import ImageSyncScanRequest
from aliyunsdkgreenextension.request.extension import HttpContentHelper
import json
import uuid
# 请使用您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret","cn-shanghai")
# 每次请求时需要新建request，请勿复用request对象。
request = ImageSyncScanRequest.ImageSyncScanRequest()
request.set_accept_format('JSON')
task = {
    "dataId": str(uuid.uuid1()),
    "url": "https://example.com/test1.jpg"
    }
# 设置待检测的图片，一张图片对应一个检测任务。
# 多张图片同时检测时，处理时间由最后一张处理完的图片决定。
# 通常情况下批量检测的平均响应时间比单任务检测长，一次批量提交的图片数越多，响应时间被拉长的概率越高。
# 代码中以单张图片检测作为示例，如果需要批量检测多张图片，请自行构建多个任务。
# 一次请求中可以检测多张图片，每张图片支持检测多个风险场景，计费按照单图片单场景检测叠加计算。
# 例如，检测2张图片，场景传递porn、terrorism，则计费按照2张图片鉴黄和2张图片暴恐检测计算。
request.set_content(HttpContentHelper.toValue({"tasks": [task], "scenes": ["porn"]}))
response = clt.do_action_with_exception(request)
print(response)
result = json.loads(response)
if 200 == result["code"]:
    taskResults = result["data"]
    for taskResult in taskResults:
        if (200 == taskResult["code"]):
            sceneResults = taskResult["results"]
            for sceneResult in sceneResults:
                scene = sceneResult["scene"]
                suggestion = sceneResult["suggestion"]
                # 根据scene和suggetion设置后续操作。
                # 根据不同的suggestion结果做业务上的不同处理。例如，将违规数据删除等。
                print(suggestion)
                print(scene)
```

- 传本地图片文件进行检测

```
# coding=utf-8
# 以下代码用于调用图片同步检测接口并实时返回检测结果。
from aliyunsdkcore import client
from aliyunsdkgreen.request.v20180509 import ImageSyncScanRequest
from aliyunsdkgreenextension.request.extension import ClientUploader
from aliyunsdkgreenextension.request.extension import HttpContentHelper
import json
import uuid
import sys
# 设置编码规则，以便支持本地中文路径。
# Python2中添加以下内容，Python3无需添加。
if sys.version_info[0] == 2:
    reload(sys)
    sys.setdefaultencoding('utf-8')
# 请使用您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", "cn-shanghai")
# 每次请求时需要新建request，请勿复用request对象。
request = ImageSyncScanRequest.ImageSyncScanRequest()
request.set_accept_format('JSON')
# 请修改成您的本地文件路径。
# 上传本地文件到服务端。
uploader = ClientUploader.getImageClientUploader(clt)
url = uploader.uploadFile('d:/test/test.jpg')
# 获取上传后的URL进行检测。
task = {"dataId": str(uuid.uuid1()),
         "url":url
        }
# 设置待检测的图片，一张图片对应一个检测任务。
# 多张图片同时检测时，处理时间由最后一张处理完的图片决定。
# 通常情况下批量检测的平均响应时间比单任务检测长，一次批量提交的图片数越多，响应时间被拉长的概率越高。
# 代码中以单张图片检测作为示例，如果需要批量检测多张图片，请自行构建多个任务。
# 一次请求中可以检测多张图片，每张图片支持检测多个风险场景，计费按照单图片单场景检测叠加计算。
# 例如，检测2张图片，场景传递porn、terrorism，则计费按照2张图片鉴黄和2张图片暴恐检测计算。
request.set_content(HttpContentHelper.toValue({"tasks": [task], "scenes": ["porn"]}))
response = clt.do_action_with_exception(request)
print(response)
result = json.loads(response)
if 200 == result["code"]:
    taskResults = result["data"]
    for taskResult in taskResults:
        if (200 == taskResult["code"]):
            sceneResults = taskResult["results"]
            for sceneResult in sceneResults:
                # 根据scene和suggetion设置后续操作。
                # 根据不同的suggestion结果做业务上的不同处理。例如，将违规数据删除等。
                scene = sceneResult["scene"]
                suggestion = sceneResult["suggestion"]
```

- 传图片二进制内容进行检测

```
# coding=utf-8
# 以下代码用于调用图片同步检测接口并实时返回检测结果。
from aliyunsdkcore import client
from aliyunsdkgreen.request.v20180509 import ImageSyncScanRequest
from aliyunsdkgreenextension.request.extension import ClientUploader
from aliyunsdkgreenextension.request.extension import HttpContentHelper
import json
import uuid
import sys
# 设置编码规则，以便支持本地中文路径。
# Python2中添加以下内容，Python3无需添加。
if sys.version_info[0] == 2:
    reload(sys)
    sys.setdefaultencoding('utf-8')
# 请使用您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret","cn-shanghai")
# 每次请求时需要新建request，请勿复用request对象。
request = ImageSyncScanRequest.ImageSyncScanRequest()
request.set_accept_format('JSON')
# 读取本地文件作为二进制数据，模拟二进制数据检测。
# 请修改成您的本地文件路径。
f = open('d:/test/test.jpg',"rb+")
imageBytes = f.read()
f.close()
# 上传二进制文件到服务端。
uploader = ClientUploader.getImageClientUploader(clt)
url = uploader.uploadBytes(imageBytes)
# 取上传后的URL进行检测。
task = {
    "dataId": str(uuid.uuid1()),
    "url":url
}
# 设置待检测的图片，一张图片对应一个检测任务。
# 多张图片同时检测时，处理时间由最后一张处理完的图片决定。
# 通常情况下批量检测的平均响应时间比单任务检测长，一次批量提交的图片数越多，响应时间被拉长的概率越高。
# 代码中以单张图片检测作为示例，如果需要批量检测多张图片，请自行构建多个任务。
# 一次请求中可以检测多张图片，每张图片支持检测多个风险场景，计费按照单图片单场景检测叠加计算。
# 例如，检测2张图片，场景传递porn、terrorism，则计费按照2张图片鉴黄和2张图片暴恐检测计算。
request.set_content(HttpContentHelper.toValue({"tasks": [task], "scenes": ["porn"]}))
response = clt.do_action_with_exception(request)
print(response)
result = json.loads(response)
if 200 == result["code"]:
    taskResults = result["data"]
    for taskResult in taskResults:
        if (200 == taskResult["code"]):
            sceneResults = taskResult["results"]
            for sceneResult in sceneResults:
                # 根据scene和suggetion设置后续操作。
                # 根据不同的suggestion结果做业务上的不同处理。例如，将违规数据删除等。
                scene = sceneResult["scene"]
                suggestion = sceneResult["suggestion"]
```

提交图片异步检测任务

使用Python SDK对图片进行风险检测，通过异步请求提交检测任务，结果可以通过提交请求时设置callback回调通知获取，也可以通过接口轮询获取。

异步检测支持的输入内容与同步检测一致（本地文件、图片URL、图片二进制流），以下仅以图片URL作为输入示例。

接口	描述	支持的地域
ImageAsyncScanRequest	提交图片异步检测任务，对图片进行多个风险场景的识别，包括色情、暴恐涉政、广告、二维码、不良场景、Logo（商标台标）识别。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
# coding=utf-8
# 以下代码用于调用图片异步检测接口，您需要根据接口返回的taskId查询检测结果。
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import ImageAsyncScanRequest
from aliyunsdkgreenextension.request.extension import HttpContentHelper
import json
import uuid
# 请使用您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret","cn-shanghai")
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
# 每次请求需要新建request，请勿复用request对象。
request = ImageAsyncScanRequest.ImageAsyncScanRequest()
request.set_accept_format('JSON')
task1 = {
    "dataId": str(uuid.uuid1()),
    "url":"http://example.com/xxx.jpg"
}
# 设置待检测的图片，一张图片对应一个检测任务。
# 多张图片同时检测时，处理时间由最后一张处理完的图片决定。
# 通常情况下批量检测的平均响应时间比单任务检测长，一次批量提交的图片数越多，响应时间被拉长的概率越高。
# 代码中以单张图片检测作为示例，如果需要批量检测多张图片，请自行构建多个任务。
# 一次请求中可以检测多张图片，每张图片支持检测多个风险场景，计费按照单图片单场景检测叠加计算。
# 例如，检测2张图片，场景传递porn、terrorism，则计费按照2张图片鉴黄和2张图片暴恐检测计算。
request.set_content(HttpContentHelper.toValue({"tasks": [task1], "scenes": ["porn"]}))
response = clt.do_action_with_exception(request)
print(response)
result = json.loads(response)
if 200 == result["code"]:
    taskResults = result["data"]
    for taskResult in taskResults:
        if(200 == taskResult["code"]):
            taskId = taskResult["taskId"]
            # 保存taskId。间隔一段时间后使用taskId查询检测结果。
            print(taskId)
```

查询异步检测结果

接口	描述	支持的地域
ImageAsyncScanResultsRequest	查询图片异步检测任务的结果。支持同时查询多个检测任务的返回结果。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
# coding=utf-8
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import ImageAsyncScanResultsRequest
from aliyunsdkgreenextension.request.extension import HttpContentHelper
import json
# 请使用您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret","cn-shanghai")
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
# 每次请求需要新建request，请勿复用request对象。
request = ImageAsyncScanResultsRequest.ImageAsyncScanResultsRequest()
request.set_accept_format('JSON')
# 通过taskId查询检测结果。
taskIds = ["img5sO$Zsssss7RYuz4Yyhhe-1q51iZ"]
request.set_content(HttpContentHelper.toValue(taskIds))
response = clt.do_action_with_exception(request)
print(response)
result = json.loads(response)
if 200 == result["code"]:
    taskResults = result["data"]
    for taskResult in taskResults:
        if (200 == taskResult["code"]):
            sceneResults = taskResult["results"]
            for sceneResult in sceneResults:
                # 根据scene和suggetion设置后续操作。
                # 根据不同的suggestion结果做业务上的不同处理。例如，将违规数据删除等。
                scene = sceneResult["scene"]
                suggestion = sceneResult["suggestion"]
```

图片检测结果反馈

如果您认为图片检测结果与您的预期不符，可以通过图片检测结果反馈接口，对检测结果进行纠正（系统会根据您反馈的结果，将图片添加到相似图片的黑名单库或者白名单库）。当您再次提交相似的内容进行检测时，以您反馈的label返回结果。

关于接口的说明，请参见检测结果反馈。

接口	描述	支持的Region

接口	描述	支持的Region
ImageScanFeedbackRequest	提交图片检测结果的反馈，以人工反馈的检测结果纠正算法检测结果。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
# coding=utf-8
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import ImageScanFeedbackRequest
import json
# 请使用您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
request = ImageScanFeedbackRequest.ImageScanFeedbackRequest()
request.set_accept_format('JSON')
# scenes：检测场景，支持指定多个场景。
# suggestion：期望的检测结果，pass：正常；block：违规。
request.set_content(json.dumps({"suggestion": "block", "scenes": ["ad", "terrorism"], "url"
: "图片链接"}))
try:
    response = clt.do_action_with_exception(request)
    print response
except Exception, err:
    print err.__str__()
```

3.3.2. 视频审核

本文介绍了如何使用Python SDK视频审核接口，检测视频中是否包含风险内容。

功能描述

视频审核接口支持同步检测和异步检测两种方式。

- 同步检测只支持传递视频的截帧图片序列。关于参数的详细介绍，请参见同步检测。
- 异步检测（推荐）支持对原始视频或视频的截帧图片序列进行检测。关于参数的详细介绍，请参见异步检测。

前提条件

- 安装Python依赖。关于安装Python依赖的具体操作，请参见安装Python依赖。

  > **说明** 请一定按照安装Python依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

（推荐）提交视频异步检测任务

接口	描述	支持的地域
VideoAsyncScanRequest	提交视频异步检测任务，对视频进行多个风险场景的识别，包括色情、暴恐涉政、广告 、不良场景、Logo（商标台标）识别。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

- 提交视频URL进行检测

```
#coding=utf-8
# 以下代码将调用视频异步检测接口。
from aliyunsdkcore import client
from aliyunsdkgreen.request.v20180509 import VideoAsyncScanRequest
from aliyunsdkgreenextension.request.extension import HttpContentHelper
import json
import uuid
# 请使用您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret","cn-shanghai")
# 每次请求时需要新建request，请勿复用request对象。
request = VideoAsyncScanRequest.VideoAsyncScanRequest()
request.set_accept_format('JSON')
task = {"dataId": str(uuid.uuid1()),
          "url": "https://www.aliyundoc.com/xxx.mp4"
        }
print(task)
# 设置待检测的视频。默认一次请求只支持检测一个视频。如果需要开放到多个，请通过工单联系我们。
# 一次请求中可以同时检测多个视频，每个视频可以同时检测多个风险场景，计费按照单视频截帧单场景检测叠加计算。
# 例如：检测2个视频，场景传入porn、terrorism，则计费按照2个视频的截帧总数*（鉴黄检测的单价+暴恐检测的单价）计算。
request.set_content(HttpContentHelper.toValue({"tasks": [task], "scenes": ["terrorism"]})
)
response = clt.do_action_with_exception(request)
print(response)
result = json.loads(response)
if 200 == result["code"]:
    taskResults = result["data"]
    for taskResult in taskResults:
        # 保存返回的taskId，用于查询视频检测的结果。
        print(taskResult["taskId"])
```

- 提交本地视频文件进行检测

```
#coding=utf-8
# 以下代码将调用视频异步检测接口。
from aliyunsdkcore import client
from aliyunsdkgreen.request.v20180509 import VideoAsyncScanRequest
from aliyunsdkgreenextension.request.extension import HttpContentHelper
from aliyunsdkgreenextension.request.extension import ClientUploader
import json
import uuid
# 设置编码规则，以便支持本地中文路径。
# Python2中添加以下内容，Python3无需添加。
if sys.version_info[0] == 2:
    reload(sys)
    sys.setdefaultencoding('utf-8')
# 请使用您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret","cn-shanghai")
# 每次请求时需要新建request，请勿复用request对象。
request = VideoAsyncScanRequest.VideoAsyncScanRequest()
request.set_accept_format('JSON')
# 上传本地文件到服务端。请修改成您自己的本地文件路径。
uploader = ClientUploader.getVideoClientUploader(clt)
url = uploader.uploadFile('d:/暴恐涉政1.mp4')
task = {"dataId": str(uuid.uuid1()),
         "url": url
        }
print(task)
# 设置待检测的视频。默认一次请求只支持检测一个视频。如果需要开放到多个，请通过工单联系我们。
# 一次请求中可以同时检测多个视频，每个视频可以同时检测多个风险场景，计费按照单视频截帧单场景检测叠加计算。
# 例如：检测2个视频，场景传入porn、terrorism，则计费按照2个视频的截帧总数*（鉴黄检测的单价+暴恐检测的单价）计算。
request.set_content(HttpContentHelper.toValue({"tasks": [task], "scenes": ["terrorism"]})
)
response = clt.do_action_with_exception(request)
print(response)
result = json.loads(response)
if 200 == result["code"]:
    taskResults = result["data"]
    for taskResult in taskResults:
        # 保存返回的taskId，用于查询视频检测的结果。
        print(taskResult["taskId"])
```

- 提交视频文件二进制文件进行检测

```
#coding=utf-8
# 以下代码将调用视频异步检测接口。
from aliyunsdkcore import client
from aliyunsdkgreen.request.v20180509 import VideoAsyncScanRequest
from aliyunsdkgreenextension.request.extension import HttpContentHelper
from aliyunsdkgreenextension.request.extension import ClientUploader
import json
import uuid
# 设置编码规则，以便支持本地中文路径。
# Python2中添加以下内容，Python3无需添加。
if sys.version_info[0] == 2:
    reload(sys)
    sys.setdefaultencoding('utf-8')
# 请使用您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret","cn-shanghai")
# 每次请求时需要新建request，请勿复用request对象。
request = VideoAsyncScanRequest.VideoAsyncScanRequest()
request.set_accept_format('JSON')
# 读取本地文件作为二进制数据，模拟二进制数据检测。
# 请修改成您自己的本地文件路径。
f = open('d:/暴恐涉政1.mp4', "rb+")
videoBytes = f.read()
f.close()
# 上传二进制文件到服务端。
uploader = ClientUploader.getVideoClientUploader(clt)
url = uploader.uploadBytes(videoBytes)
task = {"dataId": str(uuid.uuid1()),
         "url": url
        }
print(task)
# 设置待检测的视频。默认一次请求只支持检测一个视频。如果需要开放到多个，请通过工单联系我们。
# 一次请求中可以同时检测多个视频，每个视频可以同时检测多个风险场景，计费按照单视频截帧单场景检测叠加计算。
# 例如：检测2个视频，场景传入porn、terrorism，则计费按照2个视频的截帧总数*（鉴黄检测的单价+暴恐检测的单价）计算。
request.set_content(HttpContentHelper.toValue({"tasks": [task], "scenes": ["terrorism"]})
)
response = clt.do_action_with_exception(request)
print(response)
result = json.loads(response)
if 200 == result["code"]:
    taskResults = result["data"]
    for taskResult in taskResults:
        # 保存返回的taskId，用于查询视频检测的结果。
        print(taskResult["taskId"])
```

- 提交视频直播流进行检测

```
#coding=utf-8
# 以下代码将调用视频异步检测接口。
from aliyunsdkcore import client
from aliyunsdkgreen.request.v20180509 import VideoAsyncScanRequest
from aliyunsdkgreenextension.request.extension import HttpContentHelper
import json
import uuid
# 请使用您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret","cn-shanghai")
# 每次请求时需要新建request，请勿复用request对象。
request = VideoAsyncScanRequest.VideoAsyncScanRequest()
request.set_accept_format('JSON')
# 请将url替换为您的实时直播流地址。
task = {
        "dataId": str(uuid.uuid1()),
        "url": "http://www.aliyundoc.com/xxx.flv"
        }
print(task)
# 设置待检测的视频。默认一次请求只支持检测一个视频。如果需要开放到多个，请通过工单联系我们。
# 一次请求中可以同时检测多个视频，每个视频可以同时检测多个风险场景，计费按照单视频截帧单场景检测叠加计算。
# 例如：检测2个视频，场景传入porn、terrorism，则计费按照2个视频的截帧总数*（鉴黄检测的单价+暴恐检测的单价）计算。
request.set_content(HttpContentHelper.toValue({"tasks": [task], "scenes": ["terrorism"],
"live": True}))
response = clt.do_action_with_exception(request)
print(response)
result = json.loads(response)
if 200 == result["code"]:
    taskResults = result["data"]
    for taskResult in taskResults:
        # 保存返回的taskId，用于查询视频检测的结果。
        print(taskResult["taskId"])
```

- 提交视频语音进行综合检测

```
#coding=utf-8
# 以下代码将调用视频异步检测接口。
from aliyunsdkcore import client
from aliyunsdkgreen.request.v20180509 import VideoAsyncScanRequest
from aliyunsdkgreenextension.request.extension import HttpContentHelper
import json
import uuid
# 请使用您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret","cn-shanghai")
# 每次请求时需要新建request，请勿复用request对象。
request = VideoAsyncScanRequest.VideoAsyncScanRequest()
request.set_accept_format('JSON')
# 请将url替换为您的实时直播流地址。
task = {
        "dataId": str(uuid.uuid1()),
        "url": "http://www.aliyundoc.com/xxx.flv"
        }
print(task)
# 设置待检测的视频。默认一次请求只支持检测一个视频。如果需要开放到多个，请通过工单联系我们。
# 一次请求中可以同时检测多个视频，每个视频可以同时检测多个风险场景，计费按照单视频截帧单场景检测叠加计算。
# 例如：检测2个视频，场景传入porn、terrorism，则计费按照2个视频的截帧总数*（鉴黄检测的单价+暴恐检测的单价）计算。
# 如果同时检测视频画面和视频中的语音，视频中的画面按照上述示例计费，语音部分按照视频语音的总时长进行计费。
request.set_content(HttpContentHelper.toValue({"tasks": [task], "scenes": ["terrorism"],
"live": True, "audioScenes": ["antispam"]}))
response = clt.do_action_with_exception(request)
print(response)
result = json.loads(response)
if 200 == result["code"]:
    taskResults = result["data"]
    for taskResult in taskResults:
        # 保存返回的taskId，用于查询视频检测的结果。
        print(taskResult["taskId"])
```

查询视频异步检测结果

接口	描述	支持的地域
VideoAsyncScanResultsRequest	查询视频异步检测任务的结果。 **说明** 该方法需要轮询结果，建议使用callback的方式获取结果。	• **cn-shanghai**：华东2（上海） • **cn-beijing**：华北2（北京） • **cn-shenzhen**：华南1（深圳） • **ap-southeast-1**：新加坡

示例代码

```
#coding=utf-8
# 以下代码将调用视频异步检测结果查询接口。
import json
from aliyunsdkcore import client
from aliyunsdkgreen.request.v20180509 import VideoAsyncScanResultsRequest
from aliyunsdkgreenextension.request.extension import HttpContentHelper
# 请使用您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret","cn-shanghai")
# 每次请求时需要新建request，请勿复用request对象。
request = VideoAsyncScanResultsRequest.VideoAsyncScanResultsRequest()
request.set_accept_format('JSON')
# 传入需要查询的视频检测任务的taskId列表。
taskIds = ['vi3pX@vXC94hPnWsss39WOQ9-1q52ZG']
request.set_content(HttpContentHelper.toValue(taskIds))
response = clt.do_action_with_exception(request)
result = json.loads(response)
if 200 == result["code"]:
    taskResults = result["data"]
    for taskResult in taskResults:
        # 每个task的results中包含该视频的截帧检测结果。
        print(taskResult['results'])
```

提交视频截帧同步检测任务

接口	描述	支持的地域
VideoSyncScanRequest	提交视频同步检测任务，同步检测视频中的风险内容。 **说明** 同步检测只支持传递视频帧序列，不支持检测视频文件，推荐使用异步检测接口。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
#coding=utf-8
# 以下代码将调用视频同步检测接口，仅支持传入视频截帧序列进行检测。
from aliyunsdkcore import client
from aliyunsdkgreen.request.v20180509 import VideoSyncScanRequest
from aliyunsdkgreenextension.request.extension import HttpContentHelper
import json
# 请使用您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret","cn-shanghai")
# 每次请求时需要新建request，请勿复用request对象。
request = VideoSyncScanRequest.VideoSyncScanRequest()
request.set_accept_format('JSON')
task = {
            "frames":[
                {"offset" : 0, "url" : "https://example.com/test1.jpg"},
                {"offset" : 2, "url" : "https://example.com/test2.jpg"},
                {"offset" : 3, "url" : "https://example.com/test3.jpg"}
            ]
        }
print(task)
# 设置待检测的视频。默认一次请求只支持检测一个视频。如果需要开放到多个，请通过工单联系我们。
# 一次请求中可以同时检测多个视频，每个视频可以同时检测多个风险场景，计费按照单视频截帧单场景检测叠加计算。
# 例如：检测2个视频，场景传入porn、terrorism，则计费按照2个视频的截帧总数*（鉴黄检测的单价+暴恐检测的单价）计算。
request.set_content(HttpContentHelper.toValue({"tasks": [task], "scenes": ["porn"]}))
response = clt.do_action_with_exception(request)
print(response)
result = json.loads(response)
if 200 == result["code"]:
    taskResults = result["data"]
    for taskResult in taskResults:
        for result in taskResult["results"]:
            # 根据结果设置后续操作。
            print(result['suggestion'])
            print(result['scene'])
```

视频检测结果反馈

如果您认为视频检测结果与您的预期不符，可以通过视频检测结果反馈接口，对检测结果进行纠正（系统会根据您反馈的结果，将视频截帧添加到相似图片的黑名单库或者白名单库）。当您再次提交相似的内容进行检测时，以您反馈的label返回结果。

关于接口的说明，请参见检测结果反馈。

接口	描述	支持的Region
VideoFeedbackRequest	提交视频检测结果的反馈，以人工反馈的检测结果纠正算法检测结果。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
# coding=utf-8
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import VideoFeedbackRequest
import json
# 请填写您AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
request = VideoFeedbackRequest.VideoFeedbackRequest()
request.set_accept_format('JSON')
# scenes：检测场景，支持指定多个场景。
# suggestion：期望的检测结果，pass：正常；block：违规。
request.set_content(
    json.dumps({"dataId": "检测数据ID", "taskId": "视频审核任务ID", "url": "视频链接",
                "suggestion": "block", "frames": [{"url": "视频截帧链接_1", "offset": "视频截
帧offset_1"}],
                "scenes": ["ad", "terrorism"], "note": "备注信息"}))
try:
    response = clt.do_action_with_exception(request)
    print response
except Exception, err:
    print err.__str__()
```

3.3.3. 文本反垃圾检测

本文介绍了如何使用Python SDK文本反垃圾接口，对文本内容进行色情、暴恐、涉政等风险进行识别。

功能描述

文本反垃圾接口目前仅支持同步检测。关于参数的详细说明，请参见文本同步检测。

一次请求可以检测多条文本，也可以检测单条文本。按实际检测的文本条数进行计费，请参见计费方式概述。

前提条件

- 安装Python依赖。关于安装Python依赖的具体操作，请参见安装Python依赖。

  > **说明** 请一定按照安装Python依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

提交文本反垃圾检测任务

在使用文本反垃圾检测之前，您需要先提交文本内容检测任务，如果您认为文本检测的结果与您的期望不符，可以通过文本反垃圾结果反馈接口纠正算法的检测结果。文本垃圾检测支持自定义关键词，例如，添加一些竞品关键词等。如果被检测的文本中包含您添加的关键词，算法会返回suggestion为block。

您可以前往内容安全控制台添加关键词，也可以通过API接口添加关键词。

接口	描述	支持的Region

接口	描述	支持的Region
TextScanRequest	提交文本反垃圾检测任务，检测场景参数请传递antispam（scenes=antispam）。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
# coding=utf-8
# 以下代码用于调用文本检测接口。
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import TextScanRequest
import json
import uuid
import datetime
# 请使用您的AccessKey信息，支持修改配置文件或直接明文替换。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret",'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
# 每次请求时需要新建request，请勿复用request对象。
request = TextScanRequest.TextScanRequest()
request.set_accept_format('JSON')
task1 = {"dataId": str(uuid.uuid1()),
         "content":"textContentToBeModerated",
         "time":datetime.datetime.now().microsecond
        }
# 文本反垃圾检测场景的场景参数是antispam。
request.set_content(bytearray(json.dumps({"tasks": [task1], "scenes": ["antispam"]}), "utf-
8"))
response = clt.do_action_with_exception(request)
print(response)
result = json.loads(response)
if 200 == result["code"]:
    taskResults = result["data"]
    for taskResult in taskResults:
        if (200 == taskResult["code"]):
             sceneResults = taskResult["results"]
             for sceneResult in sceneResults:
                 scene = sceneResult["scene"]
                 suggestion = sceneResult["suggestion"]
                 # 根据scene和suggetion设置后续操作。
```

文本反垃圾结果反馈

如果您认为文本检测的结果与您的期望不符，您可以通过文本反垃圾结果反馈接口纠正算法的检测结果。

系统会将您反馈的结果添加到对应的检测文本库，在您下次提交相似的内容进行检测时，以您通过反馈接口校正后的结果作为检测结果。

接口	描述	支持的地域
TextFeedbackRequest	提交文本反垃圾检测结果的反馈，以人工反馈的检测结果纠正算法检测结果。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
# coding=utf-8
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import TextFeedbackRequest
import json
# 请使用您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
request = TextFeedbackRequest.TextFeedbackRequest()
request.set_accept_format('JSON')
request.set_content(
    json.dumps({"dataId": "检测数据ID", "taskId": "文本审核任务ID",
                "content": "文本内容", "label": "spam", "note": "备注信息"}))
try:
    response = clt.do_action_with_exception(request)
    print response
except Exception, err:
    print err.__str__()
```

3.3.4. 文件审核

本文介绍如何使用Python SDK文件审核接口，检测文件中的文字和图片信息。

功能描述

文件审核目前只支持异步检测（异步检测不会实时返回检测结果）任务。关于参数的详细说明，请参见提交文件检测任务。

- 支持的文件类型：
  - 支持检测以下文件中的文本内容。

    PDF、*WORD*、*TXT*、*PPT*、*EXCEL*、*OUTLOOK*、*VISIO*、*ZIP*、*TAR*、*RTF*。
  - 支持检测*PDF*文件中的图片内容。
- 支持的文件大小：5 MB以内。

前提条件

- 安装Python依赖。关于安装Python依赖的具体操作，请参见安装Python依赖。

  > **说明** 请一定按照安装Python依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

提交文件异步检测任务

接口	描述	支持的地域
FileAsyncScanRequest	提交文件异步检测任务，对文件中的文字和图片进行检测。	• **cn-shanghai**：华东2（上海） • **cn-beijing**：华北2（北京）

示例代码

```
# coding=utf-8
import uuid
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import FileAsyncScanRequest
import json
# 请使用您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
request = FileAsyncScanRequest.FileAsyncScanRequest()
request.set_accept_format('JSON')
# textScenes：检测内容包含文本时，指定检测场景，取值：antispam。imageScenes：检测内容包含图片时，指定检测场景。
request.set_content(
    json.dumps({"textScenes": ["antispam"], "imageScenes": ["porn", "ad"],
                "tasks": [
                    {"dataId": "检测数据ID",
                     "url": "待检测文件链接"}]}))
try:
    response = clt.do_action_with_exception(request)
    print response
except Exception, err:
    print err.__str__()
```

获取文件异步检测任务结果

接口	描述	支持的Region
FileAsyncScanResultsRequest	获取文件异步检测结果。	• **cn-shanghai**：华东2（上海） • **cn-beijing**：华北2（北京）

示例代码

```
# coding=utf-8
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import FileAsyncScanResultsRequest
import json
# 请使用您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
request = FileAsyncScanResultsRequest.FileAsyncScanResultsRequest()
request.set_accept_format('JSON')
request.set_content(
    json.dumps({"taskIds": ["文件检测任务ID"]}))
try:
    response = clt.do_action_with_exception(request)
    print response
except Exception, err:
    print err.__str__()
```

3.3.5. 语音反垃圾检测

本文介绍了如何使用Python SDK语音反垃圾接口，检测实时语音流或语音文件中的垃圾内容。

功能描述

语音流检测和语音文件检测均为异步检测，检测结果需要您以轮询或者回调的方式获取。关于调用请求中的检测场景参数scenes，返回结果中的分类参数label，以及操作建议参数suggestion的说明，请参见语音异步检测。

语音检测按照检测的语音文件、语音流的时间长度进行计费，计费粒度为分钟，每天累计检测总时长进行计量统计，每天检测总时长不足一分钟的按照一分钟进行计费。

前提条件

- 安装Python依赖。关于安装Python依赖的具体操作，请参见安装Python依赖。

  > **说明** 请一定按照安装Python依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

提交语音异步检测任务

接口	描述	支持的地域
VoiceAsyncScanRequest	异步检测语音流或语音文件中是否包含违规内容。语音流格式支持： • RTMP • HTTP • FLV	**cn-shanghai**：华东2（上海）

示例代码

- 提交语音URL进行检测

```
# coding=utf-8
# 以下代码用于调用语音异步检测接口。
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import VoiceAsyncScanRequest
from aliyunsdkgreenextension.request.extension import HttpContentHelper
import json
# 请使用您自己的AccessKey信息。
clt = client.AcsClient("yourAccessKeyId", "yourAccessKeySecret","cn-shanghai")
# 每次请求时需要新建request，请勿复用request对象。
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
request = VoiceAsyncScanRequest.VoiceAsyncScanRequest()
request.set_accept_format('JSON')
task = {
    "url": "https://example.com/xxxx.flv",
    }
# 如果需要检测语音流，请将live设置为true。
request.set_content(HttpContentHelper.toValue({"tasks": [task], "scenes": ["antispam"], "
live":false}))
response = clt.do_action_with_exception(request)
print(response)
result = json.loads(response)
if 200 == result["code"]:
    taskResults = result["data"]
    for taskResult in taskResults:
        code = taskResult["code"]
        if 200 == code:
            # 请保存taskId，用于查询语音检测结果。
            print(taskResult["taskId"])
```

- 提交本地语音文件进行检测

```
# coding=utf-8
# 以下代码用于调用语音异步检测接口。
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import VoiceAsyncScanRequest
from aliyunsdkgreenextension.request.extension import HttpContentHelper
from aliyunsdkgreenextension.request.extension import ClientUploader
import json
# 请使用您自己的AccessKey信息。
clt = client.AcsClient("yourAccessKeyId", "yourAccessKeySecret","cn-shanghai")
# 每次请求时需要新建request，请勿复用request对象。
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
request = VoiceAsyncScanRequest.VoiceAsyncScanRequest()
request.set_accept_format('JSON')
# 上传本地文件到服务端。请修改成您自己的本地文件路径。
uploader = ClientUploader.getVoiceClientUploader(clt)
url = uploader.uploadFile('d:/暴恐涉政1.mp3')
# 将type设置为file，表示检测语音文件。
task = {
    "url": url,
    "type": "file"
    }
request.set_content(HttpContentHelper.toValue({"tasks": [task], "scenes": ["antispam"]}))
response = clt.do_action_with_exception(request)
print(response)
result = json.loads(response)
if 200 == result["code"]:
    taskResults = result["data"]
    for taskResult in taskResults:
        code = taskResult["code"]
        if 200 == code:
            # 请保存taskId，用于查询语音检测结果。
            print(taskResult["taskId"])
```

查询语音异步检测结果

接口	描述	支持的地域
VoiceAsyncScanResultsRequest	查询异步语音检测结果。	`cn-shanghai`：华东2（上海）

示例代码

```
# coding=utf-8
# 以下代码用于调用异步语音检测结果查询接口。
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import VoiceAsyncScanResultsRequest
import json
import time
# 请使用您自己的AccessKey信息。
clt = client.AcsClient("yourAccessKeyId", "yourAccessKeySecret","cn-shanghai")
# 每次请求时需要新建request，请勿复用request对象。
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
def get_task(task_id):
    request = VoiceAsyncScanResultsRequest.VoiceAsyncScanResultsRequest()
    request.set_accept_format('JSON')
    task = {
        "taskId": task_id,
    }
    request.set_content(bytearray(json.dumps([task]), "utf-8"))
    response = clt.do_action_with_exception(request)
    result = json.loads(response)
    if 200 == result["code"]:
        taskResults = result["data"]
        for taskResult in taskResults:
            code = taskResult["code"]
            if 280 == code:
                print(taskResult["msg"])
            else:
                print(response)
            return code
    result["code"]
while True:
    code = get_task("input taskid")
    if code == 280:
        print("Scanning:")
        time.sleep(10)
    elif code == 200:
        print("=====SUCCESS=====")
        break
    else:
        print("=====ERROR=====")
        break
```

短语音同步检测

示例代码

```
# coding=utf-8
# 以下代码用于调用语音同步检测接口。
import json
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import VoiceSyncScanRequest
# 请使用您自己的AccessKey信息。
clt = client.AcsClient("您的AccessKeyId", "您的AccessKeySecret", "cn-shanghai")
# 每次请求时需要新建request，请勿复用request对象。
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
request = VoiceSyncScanRequest.VoiceSyncScanRequest()
request.set_accept_format('JSON')
# 同步语音检测耗时比较长，建议将调用超时设置为15秒。
request.set_read_timeout(15000)
task = {
    "url": "http://example.com/xxxxxxx",
}
request.set_content(json.dumps({"tasks": [task], "scenes": ["antispam"]}))
response = clt.do_action_with_exception(request)
result = json.loads(response)
print(json.dumps(result, ensure_ascii=False, indent=2))
```

3.3.6. 网页审核

本文介绍如何使用Python SDK网页审核接口，检测网页中的文字和图片信息。

功能描述

图片审核支持同步检测和异步检测两种方式。

- 同步检测实时返回检测结果。关于参数的详细信息，请参见网页同步检测。
- 异步检测需要您轮询结果或者通过callback回调通知获取检测结果。关于参数的详细信息，请参见网页异步检测。

前提条件

- 安装Python依赖。关于安装Python依赖的具体操作，请参见安装Python依赖。

  > **说明** 请一定按照安装Python依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

提交网页同步检测任务

接口	描述	支持的地域
WebpageSyncScanRequest	提交网页同步检测任务，对网页中的文字和图片进行检测。	• **cn-shanghai**：华东2（上海） • **cn-beijing**：华北2（北京）

示例代码

```
# coding=utf-8
# 以下代码用于调用文本检测接口。
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import WebpageSyncScanRequest
import json
import uuid
# 请填写您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
# 每次请求时需要新建Request，请勿复用Request对象。
request = WebpageSyncScanRequest.WebpageSyncScanRequest()
request.set_accept_format('JSON')
task1 = {"dataId": str(uuid.uuid1()),
         "url": "https://example.com/index.html"
         }
# 文本反垃圾检测场景的场景参数是antispam。
request.set_content(json.dumps({"tasks": [task1], "textScenes": ["antispam"], "imageScenes"
: ["porn"]}))
response = clt.do_action_with_exception(request)
print(response)
```

提交网页异步检测任务

接口	描述	支持的Region
WebpageAsyncScanRequest	提交网页异步检测任务，对网页中的文字和图片进行检测。	• **cn-shanghai**：华东2（上海） • **cn-beijing**：华北2（北京）

示例代码

```
# coding=utf-8
# 以下代码用于调用文本检测接口。
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import WebpageAsyncScanResultsRequest
import json
# 请填写您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret",'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
# 每次请求时需要新建Request，请勿复用Request对象。
request = WebpageAsyncScanResultsRequest.WebpageAsyncScanResultsRequest()
request.set_accept_format('JSON')
taskIds = ["img7F$1FvNKby44@N8Hxo8slO-1uRyNe"]
request.set_content(json.dumps(taskIds))
response = clt.do_action_with_exception(request)
print(response)
```

3.4. 人工审核

3.4.1. 图片人工审核

本文介绍如何使用Python SDK图片人工审核接口，进行人工审核。

功能描述

如果您认为图片检测结果（机审）与预期不符，可以使用图片人工审核接口。关于参数的详细信息，请参见图片人工审核。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Python依赖。关于安装Python依赖的具体操作，请参见安装Python依赖。

  > **说明** 请一定按照安装Python依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

提交图片人工审核任务

示例代码

```
# coding=utf-8
import uuid
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import ImageAsyncManualScanRequest
import json
# 请使用您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
request = ImageAsyncManualScanRequest.ImageAsyncManualScanRequest()
request.set_accept_format('JSON')
# callback、seed用于回调通知，可选参数。
request.set_content(
    json.dumps(
        {"tasks": [{"dataId": "检测数据ID", "url": "待检测图片链接"}], "callback": "回调地址",
         "seed": "随机字符串"}))
try:
    response = clt.do_action_with_exception(request)
    print response
except Exception, err:
    print err.__str__()
```

查询图片人工审核任务结果

示例代码

```
# coding=utf-8
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import ImageAsyncManualScanResultsRequest
import json
# 请使用您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
request = ImageAsyncManualScanResultsRequest.ImageAsyncManualScanResultsRequest()
request.set_accept_format('JSON')
request.set_content(json.dumps(["图片人工审核任务ID"]))
try:
    response = clt.do_action_with_exception(request)
    print response
except Exception, err:
    print err.__str__()
```

3.4.2. 视频人工审核

本文介绍如何使用Python SDK视频人工审核接口，进行人工审核。

功能描述

如果您认为视频检测结果（机审）与预期不符，可以使用视频人工审核接口。关于参数的详细信息，请参见视频人工审核API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Python依赖。关于安装Python依赖的具体操作，请参见安装Python依赖。

  > **说明** 请一定按照安装Python依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

提交视频人工审核任务

```
# coding=utf-8
import uuid
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import VideoAsyncManualScanRequest
import json
# 请使用您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
request = VideoAsyncManualScanRequest.VideoAsyncManualScanRequest()
request.set_accept_format('JSON')
request.set_content(json.dumps({"tasks": [{"dataId": "检测数据ID", "url": "待检测视频链接"}],
                                "callback": "回调地址",
                                "seed": "随机字符串"}))
try:
    response = clt.do_action_with_exception(request)
    print response
except Exception, err:
    print err.__str__()
```

查询视频人工审核任务结果

```
# coding=utf-8
import uuid
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import VideoAsyncManualScanResultsRequest
import json
# 请使用您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
request = VideoAsyncManualScanResultsRequest.VideoAsyncManualScanResultsRequest()
request.set_accept_format('JSON')
request.set_content(json.dumps(["视频人工审核任务ID"]))
try:
    response = clt.do_action_with_exception(request)
    print response
except Exception, err:
    print err.__str__()
```

3.4.3. 文本人工审核

本文介绍如何使用Python SDK文本人工审核接口，进行人工审核。

功能描述

如果您认为文本检测结果（机审）与预期不符，可以使用文本人工审核接口。关于参数的详细信息，请参见文本人工审核API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Python依赖。关于安装Python依赖的具体操作，请参见安装Python依赖。

  > **说明** 请一定按照安装Python依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

提交视频人工审核任务

```
# coding=utf-8
import uuid
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import TextAsyncManualScanRequest
import json
# 请使用您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
request = TextAsyncManualScanRequest.TextAsyncManualScanRequest()
request.set_accept_format('JSON')
# callback、seed用于回调通知，可选参数。
request.set_content(
    json.dumps(
        {"tasks": [{"dataId": "检测数据ID", "content": "待检测文本内容"}], "callback": "回调地
址",
         "seed": "随机字符串"}))
try:
    response = clt.do_action_with_exception(request)
    print response
except Exception, err:
    print err.__str__()
```

查询文本人工审核任务结果

```
# coding=utf-8
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import TextAsyncManualScanResultsRequest
import json
# 请使用您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
request = TextAsyncManualScanResultsRequest.TextAsyncManualScanResultsRequest()
request.set_accept_format('JSON')
request.set_content(json.dumps(["文本人工审核任务ID"]))
try:
    response = clt.do_action_with_exception(request)
    print response
except Exception, err:
    print err.__str__()
```

3.4.4. 语音人工审核

本文介绍如何使用Python SDK语音人工审核接口，进行人工审核。

功能描述

如果您认为语音检测结果（机审）与预期不符，可以使用语音人工审核。关于参数的详细信息，请参见语音人工审核API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Python依赖。关于安装Python依赖的具体操作，请参见安装Python依赖。

  > **说明** 请一定按照安装Python依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

提交语音人工审核任务

```
# coding=utf-8
import uuid
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import VoiceAsyncManualScanRequest
import json
# 请使用您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
request = VoiceAsyncManualScanRequest.VoiceAsyncManualScanRequest()
request.set_accept_format('JSON')
# callback、seed用于回调通知，可选参数。
request.set_content(
    json.dumps(
        {"tasks": [{"dataId": "检测数据ID", "url": "待检测音频链接"}], "callback": "回调地址",
         "seed": "随机字符串"}))
try:
    response = clt.do_action_with_exception(request)
    print response
except Exception, err:
    print err.__str__()
```

查询语音人工审核任务结果

```
# coding=utf-8
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import VoiceAsyncManualScanResultsRequest
import json
# 请使用您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
request = VoiceAsyncManualScanResultsRequest.VoiceAsyncManualScanResultsRequest()
request.set_accept_format('JSON')
request.set_content(json.dumps(["语音人工审核任务ID"]))
try:
    response = clt.do_action_with_exception(request)
    print response
except Exception, err:
    print err.__str__()
```

3.5. 图片OCR识别

本文介绍了如何使用Python SDK图片OCR接口，识别图片中的文字或卡证信息。

功能描述

通用OCR除了能够识别普通图片中的文字，还能识别结构化的卡证上的文字。关于参数的详细说明，请参见图片OCR检测API文档。

前提条件

- 安装Python依赖。关于安装Python依赖的具体操作，请参见安装Python依赖。

  > **说明** 请一定按照安装Python依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

提交图片同步检测任务

接口	描述	支持的Region
ImageSyncScanRequest	提交图片OCR同步识别任务，对图片中的文字进行识别（scene=ocr）。	• *cn-shanghai* • *cn-beijing* • *cn-shenzhen* • *ap-southeast-1*

示例代码

- 传图片URL进行检测

```
#coding=utf-8
# 以下代码将调用OCR图片检测接口并实时返回检测结果。
from aliyunsdkcore import client
from aliyunsdkgreen.request.v20180509 import ImageSyncScanRequest
from aliyunsdkgreenextension.request.extension import HttpContentHelper
import json
import uuid
# 请使用您的AccessKey信息。
clt = client.AcsClient("yourAccessKeyId", "yourAccessKeySecret","cn-shanghai")
# 每次请求时需要新建request，请勿复用request对象。
request = ImageSyncScanRequest.ImageSyncScanRequest()
request.set_accept_format('JSON')
task = {"dataId": str(uuid.uuid1()),
         "url":"https://example.com/test.jpg"
        }
# 卡证识别的检测类型通过extras参数设置，具体请参见API文档。
extras = {"card" : "id-card-front"}
print(task)
# 设置待检测的图片，一张图片对应一个检测任务。
# 多张图片同时检测时，处理时间由最后一张处理完的图片决定。
# 通常情况下批量检测的平均响应时间比单任务检测长，一次批量提交的图片数越多，响应时间被拉长的概率越高。
# 代码中以单张图片检测作为示例，如果需要批量检测多张图片，请自行构建多个任务。
# OCR检测按照实际检测的图片张数*检测的卡证类型单价计费。
request.set_content(HttpContentHelper.toValue({"tasks": [task],
                                               "scenes": ["ocr"],
                                               "extras": extras
                                               }))
response = clt.do_action_with_exception(request)
print(response)
result = json.loads(response)
if 200 == result["code"]:
    taskResults = result["data"]
    for taskResult in taskResults:
        if (200 == taskResult["code"]):
            sceneResults = taskResult["results"]
            print(sceneResults)
```

- 传本地图片文件进行检测

```
#coding=utf-8
# 以下代码将调用OCR图片检测接口并实时返回检测结果。
from aliyunsdkcore import client
from aliyunsdkgreen.request.v20180509 import ImageSyncScanRequest
from aliyunsdkgreenextension.request.extension import HttpContentHelper
from aliyunsdkgreenextension.request.extension import ClientUploader
import json
import uuid
# 设置编码规则，以便支持本地中文路径。
# Python2中添加以下内容，Python3无需添加。
if sys.version_info[0] == 2:
    reload(sys)
    sys.setdefaultencoding('utf-8')
# 请使用您的AccessKey信息。
clt = client.AcsClient("yourAccessKeyId", "yourAccessKeySecret","cn-shanghai")
# 每次请求时需要新建request，请勿复用request对象。
request = ImageSyncScanRequest.ImageSyncScanRequest()
request.set_accept_format('JSON')
# 上传本地文件到服务端。请修改成您的本地文件路径。
uploader = ClientUploader.getImageClientUploader(clt)
url = uploader.uploadFile('d:/test/test.jpg')
task = {"dataId": str(uuid.uuid1()),
         "url":url
        }
# 卡证识别的检测类型通过extras参数设置，具体请参见API文档。
extras = {"card" : "id-card-front"}
print(task)
# 设置待检测的图片，一张图片对应一个检测任务。
# 多张图片同时检测时，处理时间由最后一张处理完的图片决定。
# 通常情况下批量检测的平均响应时间比单任务检测长，一次批量提交的图片数越多，响应时间被拉长的概率越高。
# 代码中以单张图片检测作为示例，如果需要批量检测多张图片，请自行构建多个任务。
# OCR检测按照实际检测的图片张数*检测的卡证类型单价计费。
request.set_content(HttpContentHelper.toValue({"tasks": [task],
                                               "scenes": ["ocr"],
                                               "extras": extras
                                               }))
response = clt.do_action_with_exception(request)
print(response)
result = json.loads(response)
if 200 == result["code"]:
    taskResults = result["data"]
    for taskResult in taskResults:
        if (200 == taskResult["code"]):
            sceneResults = taskResult["results"]
            print(sceneResults)
```

- 传图片二进制字节组数据进行检测

```
#coding=utf-8
# 以下代码将调用OCR图片检测接口并实时返回检测结果。
from aliyunsdkcore import client
from aliyunsdkgreen.request.v20180509 import ImageSyncScanRequest
from aliyunsdkgreenextension.request.extension import HttpContentHelper
from aliyunsdkgreenextension.request.extension import ClientUploader
import json
import uuid
# 设置编码规则，以便支持本地中文路径。
# Python2中添加以下内容，Python3无需添加。
if sys.version_info[0] == 2:
    reload(sys)
    sys.setdefaultencoding('utf-8')
# 请使用您的AccessKey信息。
clt = client.AcsClient("yourAccessKeyId", "yourAccessKeySecret","cn-shanghai")
# 每次请求时需要新建request，请勿复用request对象。
request = ImageSyncScanRequest.ImageSyncScanRequest()
request.set_accept_format('JSON')
# 读取本地文件作为二进制数据，模拟二进制数据检测。
# 请修改成您的本地文件路径。
f = open('d:/test/test.jpg', "rb+")
imageBytes = f.read()
f.close()
# 上传二进制文件到服务端。
uploader = ClientUploader.getImageClientUploader(clt)
url = uploader.uploadBytes(imageBytes)
task = {"dataId": str(uuid.uuid1()),
         "url":url
        }
# 卡证识别的检测类型通过extras参数设置，具体请参见API文档。
extras = {"card" : "id-card-front"}
print(task)
# 设置待检测的图片，一张图片对应一个检测任务。
# 多张图片同时检测时，处理时间由最后一张处理完的图片决定。
# 通常情况下批量检测的平均响应时间比单任务检测长，一次批量提交的图片数越多，响应时间被拉长的概率越高。
# 代码中以单张图片检测作为示例，如果需要批量检测多张图片，请自行构建多个任务。
# OCR检测按照实际检测的图片张数*检测的卡证类型单价计费。
request.set_content(HttpContentHelper.toValue({"tasks": [task],
                                               "scenes": ["ocr"],
                                               "extras": extras
                                               }))
response = clt.do_action_with_exception(request)
print(response)
result = json.loads(response)
if 200 == result["code"]:
    taskResults = result["data"]
    for taskResult in taskResults:
        if (200 == taskResult["code"]):
            sceneResults = taskResult["results"]
            print(sceneResults)
```

3.6. 人脸识别

3.6.1. 人脸属性检测

本文介绍了如何使用Python SDK人脸属性检测接口，对指定的人脸图片进行属性检测。人脸属性检测仅支持以图片作为检测媒介。

功能描述

人脸属性检测能够识别图片中的人脸属性信息，包括人脸模糊度、人脸角度、人脸位置、微笑程度、是否戴眼镜、是否戴口罩、是否戴帽子、是否有胡子、是否有刘海、头发类型等。关于参数的详细说明，请参见人脸属性检测API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Python依赖。关于安装Python依赖的具体操作，请参见安装Python依赖。

  > **说明** 请一定按照安装Python依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

传入URL示例代码

```
# coding=utf-8
# 以下代码用于调用图片异步检测接口，您需要根据接口返回的TaskID查询检测结果。
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import DetectFaceRequest
import json
# 请填写您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
# 每次请求需要新建Request，请勿复用Request对象。
request = DetectFaceRequest.DetectFaceRequest()
request.set_accept_format('JSON')
request.set_content(json.dumps({"url": "http://example.com/xxx.jpg"}))
response = clt.do_action_with_exception(request)
print(response)
result = json.loads(response)
```

传入本地文件示例代码

```
# coding=utf-8
# 以下代码用于调用图片异步检测接口，您需要根据接口返回的TaskID查询检测结果。
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import DetectFaceRequest
from aliyunsdkgreenextension.request.extension import ClientUploader
import json
# 请填写您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
# 每次请求需要新建Request，请勿复用Request对象。
request = DetectFaceRequest.DetectFaceRequest()
request.set_accept_format('JSON')
clientUploader = ClientUploader.getImageClientUploader(clt)
url = clientUploader.uploadFile("/Users/Documents/tmp.jpg")
request.set_content(json.dumps({"url": url}))
response = clt.do_action_with_exception(request)
print(response)
result = json.loads(response)
```

3.6.2. 自定义人脸检索

3.6.2.1. 自定义人脸检索

本文介绍了如何使用Python SDK自定义人脸检索接口，对指定的人脸图片进行自定义人脸检索。

功能描述

图片审核支持同步检测和异步检测两种方式。

- 同步检测实时返回检测结果。关于参数的详细信息，请参见同步检测。
- 异步检测需要您轮询结果或者通过callback回调通知获取检测结果。关于参数的详细信息，请参见异步检测。

前提条件

- 安装Python依赖。关于安装Python依赖的具体操作，请参见安装Python依赖。

  > **说明** 请一定按照安装Python依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

人脸图片检索（URL）代码示例

```
# coding=utf-8
# 以下代码用于调用人脸图片（URL）检索接口并返回检测结果。
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import ImageSyncScanRequest
import json
import uuid
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
request = ImageSyncScanRequest.ImageSyncScanRequest()
request.set_accept_format('JSON')
# 同步调用支持单张图片的检索，即在指定的group中检索相似人脸。
task1 = {"dataId": str(uuid.uuid1()),
         "url": "https://example.com/tfs/xxx.jpg",
         "extras": {"groupId": "python_groupId_1"}
         }
request.set_content(bytearray(json.dumps({"tasks": [task1], "scenes": ["sface-n"]}), "utf-8
"))
response = clt.do_action_with_exception(request)
print response
result = json.loads(response)
if 200 == result["code"]:
    taskResults = result["data"]
    for taskResult in taskResults:
        if (200 == taskResult["code"]):
            sceneResults = taskResult["results"]
            for sceneResult in sceneResults:
                scene = sceneResult["scene"]
                suggestion = sceneResult["suggestion"]
                print suggestion
                print scene
                # 根据scene和suggetion做相关的处理。
```

人脸图片检索（本地文件）代码示例

```
# coding=utf-8
# 以下代码用于调用人脸图片（本地文件）检索接口并返回检测结果。
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import ImageSyncScanRequest
from aliyunsdkgreenextension.request.extension import ClientUploader
import json
import uuid
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
request = ImageSyncScanRequest.ImageSyncScanRequest()
request.set_accept_format('JSON')
# 上传本地文件到服务端。
uploader = ClientUploader.getImageClientUploader(clt)
url = uploader.uploadFile('d:/test/test.jpg')
# 同步调用支持单张图片的检索，即在指定的group中检索相似人脸。
task1 = {"dataId": str(uuid.uuid1()),
         "url": url,
         "extras": {"groupId": "python_groupId_1"}
         }
request.set_content(bytearray(json.dumps({"tasks": [task1], "scenes": ["sface-n"]}), "utf-8
"))
response = clt.do_action_with_exception(request)
print response
result = json.loads(response)
if 200 == result["code"]:
    taskResults = result["data"]
    for taskResult in taskResults:
        if (200 == taskResult["code"]):
            sceneResults = taskResult["results"]
            for sceneResult in sceneResults:
                scene = sceneResult["scene"]
                suggestion = sceneResult["suggestion"]
                print suggestion
                print scene
                # 根据scene和suggetion做相关的处理。
```

3.6.2.2. 新建个体

本文介绍了如何使用Python SDK新建个体。

功能描述

新建个体时，必须指定个体要加入的分组。关于参数的详细说明，请参见新建个体API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Python依赖。关于安装Python依赖的具体操作，请参见安装Python依赖。

> **说明** 请一定按照安装Python依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

新建个体代码示例

```
# coding=utf-8
# 新增个体。
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import AddPersonRequest
import json
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
request = AddPersonRequest.AddPersonRequest()
request.set_accept_format('JSON')
request.set_content(
    bytearray(json.dumps(
        {"personId": "python_personId_test_1", "groupIds": ["python_groupId_1"], "name": "
测试", "note": "备注"}), "utf-8"))
response = clt.do_action_with_exception(request)
print response
result = json.loads(response)
if 200 == result["code"]:
    resultObject = result["data"]
    if (200 == resultObject["code"]):
        # 200：表示添加成功。
        personId = resultObject["personId"]
        print personId
```

3.6.2.3. 设置个体

本文介绍了如何使用Python SDK设置个体信息。

功能描述

设置个体信息用于给个体添加备注信息，属于可选步骤。如果您设置了个体信息，则在自定义检索的返回结果中会包含该信息。关于参数的详细说明，请参见设置个体API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Python依赖。关于安装Python依赖的具体操作，请参见安装Python依赖。

> **说明** 请一定按照安装Python依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

设置个体代码示例

```
# coding=utf-8
# 设置个体信息。
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import SetPersonRequest
import json
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
request = SetPersonRequest.SetPersonRequest()
request.set_accept_format('JSON')
request.set_content(
    bytearray(json.dumps({"personId": "python_personId_test_1", "name": "测试", "note": "备
注"}), "utf-8"))
response = clt.do_action_with_exception(request)
print response
result = json.loads(response)
if 200 == result["code"]:
    resultObject = result["data"]
    if (200 == resultObject["code"]):
        # 200：表示添加成功。
        personId = resultObject["personId"]
        print personId
```

3.6.2.4. 获取个体信息

本文介绍了如何使用Python SDK获取个体信息。

功能描述

关于参数的详细说明，请参见查询个体信息API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Python依赖。关于安装Python依赖的具体操作，请参见安装Python依赖。

  > **说明** 请一定按照安装Python依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

获取个体信息代码示例

```
# coding=utf-8
# 获取个体信息。
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import GetPersonRequest
import json
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
request = GetPersonRequest.GetPersonRequest()
request.set_accept_format('JSON')
request.set_content(
    bytearray(json.dumps({"personId": "python_personId_test_1"}), "utf-8"))
response = clt.do_action_with_exception(request)
print response
result = json.loads(response)
if 200 == result["code"]:
    resultObject = result["data"]
    if (200 == resultObject["code"]):
        # 200：表示添加成功。
        personId = resultObject["personId"]
        print personId
```

3.6.2.5. 查询人脸信息

本文介绍了如何使用Python SDK获取人脸信息。

功能描述

根据个体ID和人脸ID查询当前个体的人脸ID以及人脸图片路径等信息。关于查询人脸的具体参数说明，请参见查询人脸图片API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Python依赖。关于安装Python依赖的具体操作，请参见安装Python依赖。

  > **说明** 请一定按照安装Python依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

查询人脸信息代码示例

```
# coding=utf-8
# 查询人脸信息。
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import GetFacesRequest
import json
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
request = GetFacesRequest.GetFacesRequest()
request.set_accept_format('JSON')
request.set_content(json.dumps({"personId": "personId_test_3", "faceIds": ["203452206484505
786"]}, "utf-8"))
response = clt.do_action_with_exception(request)
print response
```

3.6.2.6. 删除个体

本文介绍了如何使用Python SDK删除个体信息。

功能描述

删除个体时，个体对应的图片以及信息均会被删除。关于参数的详细说明，请参见删除个体API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Python依赖。关于安装Python依赖的具体操作，请参见安装Python依赖。

  > **说明** 请一定按照安装Python依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

删除个体代码示例

```
# coding=utf-8
# 删除个体。
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import DeletePersonRequest
import json
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
request = DeletePersonRequest.DeletePersonRequest()
request.set_accept_format('JSON')
request.set_content(
    bytearray(json.dumps({"personId": "python_personId_test_1"}), "utf-8"))
response = clt.do_action_with_exception(request)
print response
result = json.loads(response)
if 200 == result["code"]:
    resultObject = result["data"]
    if (200 == resultObject["code"]):
        # 200：表示添加成功。
        personId = resultObject["personId"]
        print personId
```

3.6.2.7. 新增人脸

本文介绍了如何使用Python SDK新增个体对应人脸图片。

功能描述

为个体新增一组人脸图片，一个个体最多允许包含20张人脸图片。关于参数的说明，请参见新增人脸API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Python依赖。关于安装Python依赖的具体操作，请参见安装Python依赖。

  > **说明** 请一定按照安装Python依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

新增人脸图片（URL）代码示例

```
# coding=utf-8
# 新增个体人脸，返回人脸ID。
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import AddFacesRequest
import json
# 请填写您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
request = AddFacesRequest.AddFacesRequest()
request.set_accept_format('JSON')
request.set_content(
    bytearray(json.dumps({"personId": "python_personId_test_1",
                          "urls": ["https://example.com/tfs/xxx.jpg"]}), "utf-8"))
response = clt.do_action_with_exception(request)
print response
result = json.loads(response)
if 200 == result["code"]:
    resultObject = result["data"]
    if (200 == resultObject["code"]):
        personId = resultObject["personId"]
        faceImageItems = resultObject["faceImageItems"]
        for faceImageItem in faceImageItems:
            if (faceImageItem["success"]):
                # success为true表示添加成功。
                print faceImageItem["faceId"]
```

新增人脸图片（本地文件）代码示例

```
# coding=utf-8
# 个体新增人脸，返回人脸Id
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import AddFacesRequest
from aliyunsdkgreenextension.request.extension import ClientUploader
import json
# 请填写您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
request = AddFacesRequest.AddFacesRequest()
request.set_accept_format('JSON')
# 上传本地文件到服务端。
uploader = ClientUploader.getImageClientUploader(clt)
url = uploader.uploadFile('d:/test/test.jpg')
request.set_content(
    bytearray(json.dumps({"personId": "python_personId_test_1",
                          "urls": [url]}), "utf-8"))
response = clt.do_action_with_exception(request)
print response
result = json.loads(response)
if 200 == result["code"]:
    resultObject = result["data"]
    if (200 == resultObject["code"]):
        personId = resultObject["personId"]
        faceImageItems = resultObject["faceImageItems"]
        for faceImageItem in faceImageItems:
            if (faceImageItem["success"]):
                # success为true表示添加成功。
                print faceImageItem["faceId"]
```

3.6.2.8. 删除人脸

本文介绍了如何使用Python SDK删除个体对应的人脸图片。

功能描述

删除人脸图片时，必须指定要删除的图片ID以及对应的个体ID信息。关于参数的详细说明，请参见删除人脸API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Python依赖。关于安装Python依赖的具体操作，请参见安装Python依赖。

  > **说明** 请一定按照安装Python依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

删除人脸代码示例

```
# coding=utf-8
# 删除人脸。
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import DeleteFacesRequest
import json
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
request = DeleteFacesRequest.DeleteFacesRequest()
request.set_accept_format('JSON')
request.set_content(
    bytearray(json.dumps({"personId": "python_personId_test_1", "faceIds": ["31842550077981
745"]}), "utf-8"))
response = clt.do_action_with_exception(request)
print response
result = json.loads(response)
if 200 == result["code"]:
    resultObject = result["data"]
    if (200 == resultObject["code"]):
        personId = resultObject["personId"]
        print personId
```

3.6.2.9. 获取组列表

本文介绍了如何使用Python SDK获取所有人脸分组ID。

功能描述

查询所有的个体组ID。关于参数的详细说明，请参见查询组列表API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Python依赖。关于安装Python依赖的具体操作，请参见安装Python依赖。

  > **说明** 请一定按照安装Python依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

获取所有分组ID代码示例

```
# coding=utf-8
# 获取所有分组ID。
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import GetGroupsRequest
import json
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
request = GetGroupsRequest.GetGroupsRequest()
request.set_accept_format('JSON')
response = clt.do_action(request)
print response
result = json.loads(response)
if 200 == result["code"]:
    resultObject = result["data"]
    if (200 == resultObject["code"]):
        # 200表示添加成功。
        groups = resultObject["groupIds"]
        print groups
```

3.6.2.10. 获取个体列表

本文介绍了如何使用Python SDK获取个体列表信息。

功能描述

每个个体必须属于一个分组，在调用该接口时指定某个分组，则返回该分组下的所有个体ID列表。关于参数的详细说明，请参见查询个体列表API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Python依赖。关于安装Python依赖的具体操作，请参见安装Python依赖。

  > **说明** 请一定按照安装Python依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

获取个体列表代码示例

```
# coding=utf-8
# 获取指定组下的个体列表。
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import GetPersonsRequest
import json
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
request = GetPersonsRequest.GetPersonsRequest()
request.set_accept_format('JSON')
request.set_content(
    bytearray(json.dumps({"groupId": "python_groupId_1"}), "utf-8"))
response = clt.do_action_with_exception(request)
print response
result = json.loads(response)
if 200 == result["code"]:
    resultObject = result["data"]
    if (200 == resultObject["code"]):
        # 200表示添加成功。
        personIds = resultObject["personIds"]
        print personIds
```

3.6.2.11. 添加个体到分组

本文介绍了如何使用Python SDK添加个体到指定分组。

功能描述

将一个个体添加到一个或者多个个体组。关于参数的详细说明，请参见添加个体到分组API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Python依赖。关于安装Python依赖的具体操作，请参见安装Python依赖。

  > **说明** 请一定按照安装Python依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

添加个体到分组代码示例

```
# coding=utf-8
# 将个体增加到指定组。
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import AddGroupsRequest
import json
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
request = AddGroupsRequest.AddGroupsRequest()
request.set_accept_format('JSON')
request.set_content(
    bytearray(json.dumps({"personId": "python_personId_test_1", "groupIds": ["nantuan"]}),
"utf-8"))
response = clt.do_action_with_exception(request)
print response
result = json.loads(response)
if 200 == result["code"]:
    resultObject = result["data"]
    if (200 == resultObject["code"]):
        # 200表示添加成功。
        personId = resultObject["personId"]
        print personId
```

3.6.2.12. 删除分组中个体

本文介绍了如何使用Python SDK从指定分组中删除个体信息。

功能描述

删除分组中个体不会删除个体对应的信息以及图片，只是解除个体与该分组的绑定关系。关于参数的详细说明，请参见删除分组中的个体API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Python依赖。关于安装Python依赖的具体操作，请参见安装Python依赖。

  > **说明** 请一定按照安装Python依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

从指定分组删除个体代码示例

```
# coding=utf-8
# 将个体从指定组移除。
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import DeleteGroupsRequest
import json
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
request = DeleteGroupsRequest.DeleteGroupsRequest()
request.set_accept_format('JSON')
request.set_content(
    bytearray(json.dumps({"personId": "python_personId_test_1", "groupIds": ["python_groupI
d_1"]}), "utf-8"))
response = clt.do_action_with_exception(request)
print response
result = json.loads(response)
if 200 == result["code"]:
    resultObject = result["data"]
    if (200 == resultObject["code"]):
        personId = resultObject["personId"]
        print personId
```

3.7. 其他

3.7.1. 相似图检索

3.7.1.1. 创建相似图样本库

本文介绍了如何使用Python SDK，创建相似图样本库。

功能描述

通过该接口创建相似图样本图库。关于参数的详细说明，请参见创建图库API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Python依赖。关于安装Python依赖的具体操作，请参见安装Python依赖。

  > **说明** 请一定按照安装Python依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

创建相似图样本任务

```
# coding=utf-8
import json
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import AddSimilarityLibraryRequest
# 请使用您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
# 每次请求时需要新建request，请勿复用request对象。
request = AddSimilarityLibraryRequest.AddSimilarityLibraryRequest()
request.set_accept_format('JSON')
request.set_content(json.dumps({"name": "相似图样本图库名称"}))
try:
    response = clt.do_action_with_exception(request)
    print response
except Exception, err:
    print err.__str__()
```

3.7.1.2. 相似图检索

本文介绍了如何使用Python SDK，对指定的图片进行相似图检索的示例代码。

功能描述

相似图检索帮助您从图库中检索出与给定的图片相似的若干张图片。关于参数的详细说明，请参见相似图检索API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Python依赖。关于安装Python依赖的具体操作，请参见安装Python依赖。

  > **说明** 请一定按照安装Python依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

传入URL示例代码

```
# coding=utf-8
# 以下代码用于调用图片同步检测接口并实时返回检测结果。
from aliyunsdkcore import client
from aliyunsdkgreen.request.v20180509 import ImageSyncScanRequest
import json
import uuid
# 请填写您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
# 每次请求时需要新建Request，请勿复用Request对象。
request = ImageSyncScanRequest.ImageSyncScanRequest()
request.set_accept_format('JSON')
task = {
    "dataId": str(uuid.uuid1()),
    "url": "https://example.com/tfs/xxx-600-600.jpg"
}
# similarity表示进行相似图检索。
request.set_content(json.dumps({"tasks": [task], "scenes": ["similarity"]}))
response = clt.do_action_with_exception(request)
print(response)
```

传入本地文件示例代码

```
# coding=utf-8
# 以下代码用于调用图片同步检测接口并实时返回检测结果。
from aliyunsdkcore import client
from aliyunsdkgreen.request.v20180509 import ImageSyncScanRequest
from aliyunsdkgreenextension.request.extension import ClientUploader
import json
import uuid
# 请使用您自己的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
# 每次请求时需要新建Request，请勿复用Request对象。
request = ImageSyncScanRequest.ImageSyncScanRequest()
request.set_accept_format('JSON')
clientUploader = ClientUploader.getImageClientUploader(clt)
url = clientUploader.uploadFile("/Users/Documents/tmp.jpg")
task = {
    "dataId": str(uuid.uuid1()),
    "url": url
}
# similarity表示进行相似图检索。
request.set_content(json.dumps({"tasks": [task], "scenes": ["similarity"]}))
response = clt.do_action_with_exception(request)
print(response)
```

3.7.1.3. 增加相似图样本

本文介绍了如何使用Python SDK，将指定的相似图图片添加到对应图库。

功能描述

添加相似图样本时，您可以为图片设置标签。如果样本图片在检索时被命中，其标签信息也会被返回。关于参数的详细说明，请参见增加样本图片API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Python依赖。关于安装Python依赖的具体操作，请参见安装Python依赖。

  > **说明** 请一定按照安装Python依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

传入URL示例代码

```
# coding=utf-8
import json
import uuid
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import AddSimilarityImageRequest
# 请填写您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
# 每次请求时需要新建Request，请勿复用Request对象。
request = AddSimilarityImageRequest.AddSimilarityImageRequest()
request.set_accept_format('JSON')
task = {
    "dataId": str(uuid.uuid1()),
    "url": "https://example.com/test1.jpg",
    "tag": ["自定义标签1", "自定义标签2", "自定义标签3"]
}
request.set_content(json.dumps({"tasks": [task]}))
response = clt.do_action_with_exception(request)
print(response)
```

传入本地文件示例代码

```
# coding=utf-8
import json
import uuid
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import AddSimilarityImageRequest
from aliyunsdkgreenextension.request.extension import ClientUploader
# 请填写您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
# 每次请求时需要新建Request，请勿复用Request对象。
request = AddSimilarityImageRequest.AddSimilarityImageRequest()
request.set_accept_format('JSON')
clientUploader = ClientUploader.getImageClientUploader(clt)
url = clientUploader.uploadFile("/Users/Documents/tmp.jpg")
task = {
    "dataId": str(uuid.uuid1()),
    "url": url,
    "tag": ["自定义标签1", "自定义标签2", "自定义标签3"]
}
request.set_content(json.dumps({"tasks": [task]}))
response = clt.do_action_with_exception(request)
print(response)
```

3.7.1.4. 查询相似图库列表

本文介绍了如何使用Python SDK，分页查询相似图库及其元素信息。

功能描述

调用该接口分页查询所有相似图库及其元素信息。关于参数的详细说明，请参见查询相似图库列表API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Python依赖。关于安装Python依赖的具体操作，请参见安装Python依赖。

  > **说明** 请一定按照安装Python依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

查询相似图库列表任务

```
# coding=utf-8
import json
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import ListSimilarityLibrariesRequest
# 请使用您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
# 每次请求时需要新建request，请勿复用request对象。
request = ListSimilarityLibrariesRequest.ListSimilarityLibrariesRequest()
request.set_accept_format('JSON')
# pageSize:每页大小，取值范围：(0,50]；currentPage：当前页码，取值范围：(0,50]
request.set_content(json.dumps({"pageSize": "5", "currentPage": "1"}))
try:
    response = clt.do_action_with_exception(request)
    print response
except Exception, err:
    print err.__str__()
```

3.7.1.5. 查询指定相似图库信息

本文介绍了如何使用Python SDK，查询指定相似图库信息。

功能描述

调用该接口查询指定相似图库信息。关于参数的详细说明，请参见查询指定相似图库API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Python依赖。关于安装Python依赖的具体操作，请参见安装Python依赖。

  > **说明** 请一定按照安装Python依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

查询指定相似图库任务

```
# coding=utf-8
import json
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import GetSimilarityLibraryRequest
# 请使用您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
# 每次请求时需要新建request，请勿复用request对象。
request = GetSimilarityLibraryRequest.GetSimilarityLibraryRequest()
request.set_accept_format('JSON')
request.set_content(json.dumps({"name": "相似图样本图库名称"}))
try:
    response = clt.do_action_with_exception(request)
    print response
except Exception, err:
    print err.__str__()
```

3.7.1.6. 查询相似图样本图片列表

本文介绍了如何使用Python SDK，分页查询指定相似图库下的所有样本图片信息。

功能描述

调用该接口分页查询指定相似图库下的所有样本图片信息。关于参数的详细说明，请参见查询样本图片列表API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Python依赖。关于安装Python依赖的具体操作，请参见安装Python依赖。

  > **说明** 请一定按照安装Python依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

查询相似图样本图片列表任务

```
# coding=utf-8
import json
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import ListSimilarityImagesRequest
# 请使用您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
# 每次请求时需要新建request，请勿复用request对象。
request = ListSimilarityImagesRequest.ListSimilarityImagesRequest()
request.set_accept_format('JSON')
# pageSize：每页大小，取值范围：(0,50]；currentPage：当前页码，取值范围：(0,50]
request.set_content(json.dumps({"library": "相似图样本图库名称", "pageSize": "5", "currentPag
e": "1"}))
try:
    response = clt.do_action_with_exception(request)
    print response
except Exception, err:
    print err.__str__()
```

3.7.1.7. 查询相似图样本图片详情

本文介绍了如何使用Python SDK，查询相似图样本图片详情。

功能描述

调用该接口查询相似图样本图片详情。关于参数的详细说明，请参见查询样本图库详情API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Python依赖。关于安装Python依赖的具体操作，请参见安装Python依赖。

  > **说明** 请一定按照安装Python依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

查询相似图样本图片详情任务

```
# coding=utf-8
import json
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import GetSimilarityImageRequest
# 请使用您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
# 每次请求时需要新建request，请勿复用request对象。
request = GetSimilarityImageRequest.GetSimilarityImageRequest()
request.set_accept_format('JSON')
request.set_content(json.dumps({"library": "相似图样本图库名称", "dataId": "样本图片ID"}))
try:
    response = clt.do_action_with_exception(request)
    print response
except Exception, err:
    print err.__str__()
```

3.7.1.8. 移除相似图样本

本文介绍了如何使用Python SDK，将相似图样本从图库中移除。

功能描述

移除操作在1分钟之内生效。一次最多允许移除100张样本图片。关于参数的详细说明，请参见删除样本图片API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Python依赖。关于安装Python依赖的具体操作，请参见安装Python依赖。

  > **说明** 请一定按照安装Python依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

示例代码

```
# coding=utf-8
import json
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import DeleteSimilarityImageRequest
# 请填写您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
# 每次请求时需要新建Request，请勿复用Request对象。
request = DeleteSimilarityImageRequest.DeleteSimilarityImageRequest()
request.set_accept_format('JSON')
request.set_content(json.dumps({"dataIds": ["x-3340-4b10-a622-7a5668f24b81"]}))
response = clt.do_action_with_exception(request)
print(response)
```

3.7.1.9. 删除相似图库

本文介绍了如何使用Python SDK，删除指定相似图库。

功能描述

只有当相似图库中无图片样本时，才允许被删除。关于参数的详细说明，请参见删除图库API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Python依赖。关于安装Python依赖的具体操作，请参见安装Python依赖。

  > **说明** 请一定按照安装Python依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

删除相似图库任务

```
# coding=utf-8
import json
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20180509 import DeleteSimilarityLibraryRequest
# 请使用您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
# 每次请求时需要新建request，请勿复用request对象。
request = DeleteSimilarityLibraryRequest.DeleteSimilarityLibraryRequest()
request.set_accept_format('JSON')
request.set_content(json.dumps({"name": "相似图样本图库名称"}))
try:
    response = clt.do_action_with_exception(request)
    print response
except Exception, err:
    print err.__str__()
```

3.7.2. 自定义文本库

本文介绍了如何使用Python SDK管理自定义文本库，以满足文本反垃圾检测场景的个性化需求。

功能描述

根据文本类型的不同，文本库分为关键词文本库和相似文本库；根据管控目的不同，文本库分为白名单、黑名单、疑似名单。关于参数的详细信息，请参见自定义文本库API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Python依赖。关于安装Python依赖的具体操作，请参见安装Python依赖。

  > **说明** 请一定按照安装Python依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

获取文本库列表

- 获取关键词文本库列表

```
# coding=utf-8
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20170823 import DescribeKeywordLibRequest
import json
# 请填写您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
# 每次请求时需要新建Request，请勿复用Request对象。
request = DescribeKeywordLibRequest.DescribeKeywordLibRequest()
request.set_ServiceModule("open_api")
response = clt.do_action_with_exception(request)
print(response)
result = json.loads(response)
allLibs = result["KeywordLibList"]
textAntispamKeywordLibs = []
for keywordLib in allLibs:
    libType = keywordLib["LibType"]
    resourceType = keywordLib["ResourceType"]
    source = keywordLib["Source"]
    # 获取文本反垃圾自定义的关键词文本库。
    if "textKeyword" == libType and "TEXT" == resourceType and "MANUAL" == source:
        textAntispamKeywordLibs.append(keywordLib)
    # 获取文本反垃圾回流的关键词文本库。
    if "textKeyword" == libType and "TEXT" == resourceType and "FEEDBACK" == source:
        textAntispamKeywordLibs.append(keywordLib)
print (json.dumps(textAntispamKeywordLibs))
```

- 获取相似文本文本库列表（包含自定义和系统回流的相似文本文本库）

```
# coding=utf-8
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20170823 import DescribeKeywordLibRequest
import json
# 请填写您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
# 每次请求时需要新建Request，请勿复用Request对象。
request = DescribeKeywordLibRequest.DescribeKeywordLibRequest()
request.set_ServiceModule("open_api")
response = clt.do_action_with_exception(request)
print(response)
result = json.loads(response)
allLibs = result["KeywordLibList"]
similarTextLibs = []
for keywordLib in allLibs:
    libType = keywordLib["LibType"]
    resourceType = keywordLib["ResourceType"]
    source = keywordLib["Source"]
    # 获取文本反垃圾自定义的相似文本文本库。
    if "similarText" == libType and "TEXT" == resourceType and "MANUAL" == source:
        similarTextLibs.append(keywordLib)
    # 获取文本反垃圾回流的相似文本文本库。
    if "similarText" == libType and "TEXT" == resourceType and "FEEDBACK" == source:
        similarTextLibs.append(keywordLib)
print (json.dumps(similarTextLibs))
```

创建文本库

- 创建关键词文本库

```
# coding=utf-8
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20170823 import CreateKeywordLibRequest
# 请填写您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
# 每次请求时需要新建Request，请勿复用Request对象。
request = CreateKeywordLibRequest.CreateKeywordLibRequest()
request.set_ServiceModule("open_api")
request.set_Name("测试关键词文本库")
# 设置为文本反垃圾场景使用。
request.set_ResourceType("TEXT")
# 设置类型为关键词。
request.set_LibType("textKeyword")
# 设置创建黑库。
request.set_Category("BLACK")
response = clt.do_action_with_exception(request)
print(response)
```

- 创建相似文本文本库

```
# coding=utf-8
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20170823 import CreateKeywordLibRequest
# 请填写您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
# 每次请求时需要新建Request，请勿复用Request对象。
request = CreateKeywordLibRequest.CreateKeywordLibRequest()
request.set_ServiceModule("open_api")
request.set_Name("测试相似文本库")
# 设置为文本反垃圾场景使用。
request.set_ResourceType("TEXT")
# 设置类型为相似文本。
request.set_LibType("similarText")
# 设置创建黑库。
request.set_Category("BLACK")
response = clt.do_action_with_exception(request)
print(response)
```

修改文本库

更新文本库库名以及修改文本库所适用的业务场景（BizType）。

```
# coding=utf-8
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20170823 import UpdateKeywordLibRequest
# 请填写您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret",'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
# 每次请求时需要新建Request，请勿复用Request对象。
request = UpdateKeywordLibRequest.UpdateKeywordLibRequest()
request.set_Id(7534001)
request.set_Name('测试修改名称')
response = clt.do_action_with_exception(request)
print(response)
```

删除文本库

> **注意** 删除文本库也将删除文本库下的文本。系统回流的文本库不允许删除。

```
# coding=utf-8
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20170823 import DeleteKeywordLibRequest
# 请填写您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
# 每次请求时需要新建Request，请勿复用Request对象。
request = DeleteKeywordLibRequest.DeleteKeywordLibRequest()
request.set_Id(3353)
response = clt.do_action_with_exception(request)
print(response)
```

查找文本

默认分页获取所有文本。如果设置了Keyword字段，将模糊查找包含该字段值的文本。

```
# coding=utf-8
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20170823 import DescribeKeywordRequest
# 请使用您自己的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
# 每次请求时需要新建Request，请勿复用Request对象。
request = DescribeKeywordRequest.DescribeKeywordRequest()
request.set_KeywordLibId("2693")
request.set_PageSize(10)
request.set_CurrentPage(1)
response = clt.do_action_with_exception(request)
print(response)
```

添加文本

```
# coding=utf-8
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20170823 import CreateKeywordRequest
# 请填写您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
# 每次请求时需要新建Request，请勿复用Request对象。
request = CreateKeywordRequest.CreateKeywordRequest()
request.set_KeywordLibId(2693)
request.set_Keywords("['你好吗', '棒棒的']")
response = clt.do_action_with_exception(request)
print(response)
```

删除文本

```
# coding=utf-8
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20170823 import DeleteKeywordRequest
# 请填写您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret",'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
# 每次请求时需要新建Request，请勿复用Request对象。
request = DeleteKeywordRequest.DeleteKeywordRequest()
request.set_KeywordLibId(2693)
request.set_Ids("[1,2]")
response = clt.do_action_with_exception(request)
print(response)
```

3.7.3. 自定义图库

本文介绍了如何使用Python SDK管理自定义图库。

功能描述

您可以自定义智能鉴黄、暴恐涉政识别、图片或视频广告的图片样本，满足个性化内容管控需求。关于参数的详细信息，请参见创建图库API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

- 安装Python依赖。关于安装Python依赖的具体操作，请参见安装Python依赖。

  > **说明** 请一定按照安装Python依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

获取自定义图库列表

您可以使用以下代码获取用户图库列表（包括用户自定义的图库列表和系统回流图库）：

```
# coding=utf-8
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20170823 import DescribeImageLibRequest
# 请填写您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
# 每次请求时需要新建Request，请勿复用Request对象。
request = DescribeImageLibRequest.DescribeImageLibRequest()
request.set_ServiceModule("open_api")
response = clt.do_action_with_exception(request)
print(response)
```

创建自定义图库

您可以使用以下代码创建自定义图库：

> **说明** 请根据您所需要的业务场景设置不同的参数。

```
# coding=utf-8
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20170823 import CreateImageLibRequest
# 请填写您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
# 每次请求时需要新建Request，请勿复用Request对象。
request = CreateImageLibRequest.CreateImageLibRequest()
request.set_ServiceModule("open_api")
request.set_Name("鉴黄黑库")
request.set_Scene("PORN")
request.set_Category("BLACK")
response = clt.do_action_with_exception(request)
print(response)
```

修改自定义图库

您可以使用以下代码修改自定义图库的名称及其适用的业务场景（BizType）：

```
# coding=utf-8
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20170823 import UpdateImageLibRequest
# 请填写您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
# 每次请求时需要新建Request，请勿复用Request对象。
request = UpdateImageLibRequest.UpdateImageLibRequest()
request.set_Id(12345)
request.set_Name("鉴黄黑库改名")
request.set_Scene("PORN")
request.set_Category("WHITE")
response = clt.do_action_with_exception(request)
print(response)
```

删除自定义图库

您可以使用以下代码删除自定义图库：

> **说明** 删除自定义图库时，图库下的所有图片也将被删除。

```
# coding=utf-8
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20170823 import DeleteImageLibRequest
# 请填写您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
# 每次请求时需要新建Request，请勿复用Request对象。
request = DeleteImageLibRequest.DeleteImageLibRequest()
request.set_Id(12345)
response = clt.do_action_with_exception(request)
print(response)
```

获取自定义图库图片列表

您可以使用以下代码获取自定义图库中所有已添加的图片列表：

```
# coding=utf-8
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20170823 import DescribeImageFromLibRequest
# 请填写您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
# 每次请求时需要新建Request，请勿复用Request对象。
request = DescribeImageFromLibRequest.DescribeImageFromLibRequest()
request.set_ImageLibId(12345)
request.set_PageSize(10)
request.set_CurrentPage(1)
response = clt.do_action_with_exception(request)
print(response)
```

删除自定义图片

您可以使用以下代码删除自定义图库中的多张自定义图片：

```
# coding=utf-8
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20170823 import DeleteImageFromLibRequest
# 请填写您的AccessKey信息。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', 'cn-shanghai', 'green.cn-shanghai.aliyuncs.com')
# 每次请求时需要新建Request，请勿复用Request对象。
request = DeleteImageFromLibRequest.DeleteImageFromLibRequest()
request.set_Ids("['669310']")
response = clt.do_action_with_exception(request)
print(response)
```

3.7.4. OSS违规检测

本文介绍了如何使用Python SDK进行OSS违规检测。

前提条件

- 安装Python依赖。关于安装Python依赖的具体操作，请参见安装Python依赖。

  > **说明** 请一定按照安装Python依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

查询OSS中的违规数据

> **说明** 以下代码仅为简单示例，具体的接口参数请参见OSS内容检测API。

```
#coding=utf-8
from aliyunsdkcore import client
from aliyunsdkgreen.request.v20170823 import DescribeOssResultItemsRequest
# 请使用您自己的AccessKey信息。您可以将其配置在配置文件里面，也可以直接明文替换。
clt = client.AcsClient("yourAccessKeyId", "yourAccessKeySecret","cn-shanghai")
# 每次请求时需要新建request，请勿复用request对象。
request = DescribeOssResultItemsRequest.DescribeOssResultItemsRequest()
request.set_ResourceType("VIDEO")
request.set_Scene("porn")
request.set_Stock("false")
request.set_StartDate("2019-01-02 00:00:00 +0800")
request.set_EndDate("2019-01-07 15:00:53 +0800")
response = clt.do_action_with_exception(request)
print(response)
```

对OSS的审核结果进行标记

如果需要对已检测出结果的内容执行删除、标记为正常并忽略，或者解除冻结等操作，您可以调用本接口。

> **说明** 以下代码仅为简单示例，具体的接口参数请参见OSS内容检测API。

```
#coding=utf-8
from aliyunsdkcore import client
from aliyunsdkgreen.request.v20170823 import MarkOssResultRequest
import json
# 请使用您自己的AccessKey信息。您可以将其配置在配置文件里面，也可以直接明文替换。
clt = client.AcsClient("yourAccessKeyId", "yourAccessKeySecret","cn-shanghai")
# 每次请求时需要新建request，请勿复用request对象。
request = MarkOssResultRequest.MarkOssResultRequest()
request.set_ResourceType("VIDEO")
request.set_Scene("porn")
request.set_Stock("false")
request.set_Ids(json.dumps([10001]))
request.set_Operation("ignore")
response = clt.do_action_with_exception(request)
print(response)
```

以TXT文件形式导出OSS违规检测结果

每次最多导出5,000条记录。导出的文件以TXT形式存储，每行一条记录。每条记录包括以下内容：OSS扫描的Bucket名称，OSS扫描的文件（key），扫描的分数结果，扫描的处理建议（suggestion）。

> **说明** 以下代码仅为简单示例，关于接口的参数描述，请参见OSS内容检测API文档。

```
#coding=utf-8
from aliyunsdkcore import client
from aliyunsdkgreen.request.v20170823 import ExportOssResultRequest
import json
# 请使用您自己的AccessKey信息。您可以将其配置在配置文件里面，也可以直接明文替换。
clt = client.AcsClient("yourAccessKeyId", "yourAccessKeySecret","cn-shanghai")
# 每次请求时需要新建request，请勿复用request对象。
request = ExportOssResultRequest.ExportOssResultRequest()
request.set_ResourceType("VIDEO")
request.set_Scene("porn")
request.set_Stock("false")
request.set_StartDate("2019-01-02 00:00:00 +0800")
request.set_EndDate("2019-01-07 15:00:53 +0800")
response = clt.do_action_with_exception(request)
print(response)
```

3.7.5. 业务场景管理

3.7.5.1. 创建业务场景

本文介绍了如何通过Python SDK创建业务场景，业务场景主要用于内容检测API功能自定义机审标准。

前提条件

- 安装Python依赖。关于安装Python依赖的具体操作，请参见安装Python依赖。

  > **说明** 请一定按照安装Python依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

提交创建业务场景任务

接口	描述	支持的地域
CreateBizType	创建业务场景。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
# coding=utf-8
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20170823 import CreateBizTypeRequest
# 请替换成您的AccessKey ID、AccessKey Secret、Region ID。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", 'cn-shanghai')
region_provider.modify_point('Green', "cn-shanghai", 'green.cn-shanghai.aliyuncs.com')
request = CreateBizTypeRequest.CreateBizTypeRequest()
# 创建业务场景的名称。
request.set_BizTypeName("您要创建的业务场景")
# 导入已经创建的BizType对应的配置。可以不设置。
request.set_BizTypeImport("您已经创建的业务场景")
# 是否引入行业模板配置，当值为true时必须传industryInfo。可以不设置。
request.set_CiteTemplate(True)
# 行业分类。CiteTemplate为true时，必须设置。取值范围：社交-注册信息-头像；社交-注册信息-昵称。
request.set_IndustryInfo("社交-注册信息-头像")
request.set_accept_format('JSON')
response = clt.do_action_with_exception(request)
print(response)
```

3.7.5.2. 删除业务场景

本文介绍了如何通过Python SDK删除业务场景。如果您需要删除业务场景，请务必保证业务场景下无数据调用再删除，否则删除业务场景后对应的策略配置恢复为默认，有可能导致部分业务数据识别错误。

前提条件

- 安装Python依赖。关于安装Python依赖的具体操作，请参见安装Python依赖。

  > **说明** 请一定按照安装Python依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

提交删除业务场景任务

接口	描述	支持的地域
DeleteBizType	删除业务场景。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
# coding=utf-8
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20170823 import DeleteBizTypeRequest
# 请替换成您的AccessKey ID、AccessKey Secret、Region ID。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", "cn-shanghai")
region_provider.modify_point('Green', "cn-shanghai", 'green.cn-shanghai.aliyuncs.com')
request = DeleteBizTypeRequest.DeleteBizTypeRequest()
# 设置要删除的业务场景。
request.set_BizTypeName("您要删除的业务场景")
request.set_accept_format('JSON')
response = clt.do_action_with_exception(request)
print(response)
```

3.7.5.3. 查询业务场景

文本介绍了如何通过Python SDK查询已创建和自定义的业务场景列表，用于在后台管理业务场景数据。

前提条件

- 安装Python依赖。关于安装Python依赖的具体操作，请参见安装Python依赖。

  > **说明** 请一定按照安装Python依赖页面中的版本安装，否则会导致调用失败。

- 如果使用本地文件或者二进制文件检测，请下载并在项目工程中引入Extension.Uploader工具类。

提交查询业务场景任务

接口	描述	支持的地域
DescribeUserBizTypes	查询业务场景。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例

```
# coding=utf-8
from aliyunsdkcore import client
from aliyunsdkcore.profile import region_provider
from aliyunsdkgreen.request.v20170823 import DescribeUserBizTypesRequest
# 请替换成您的AccessKey ID、AccessKey Secret、Region ID。
clt = client.AcsClient("您的AccessKey ID", "您的AccessKey Secret", "cn-shanghai")
region_provider.modify_point('Green', "cn-shanghai", 'green.cn-shanghai.aliyuncs.com')
request = DescribeUserBizTypesRequest.DescribeUserBizTypesRequest()
request.set_accept_format('JSON')
response = clt.do_action_with_exception(request)
'''
返回结果说明:
{
    # 所有业务场景列表。
    "BizTypeList":[
        {
            # 是否属于引用行业模板。
            "CiteTemplate":false,
            # 业务场景名称。
            "BizType":"mohongtest2",
            # 业务场景来源：用户自定义：custom。安全服务默认配置：system。
            "Source":"custom",
            # 行业信息。
            "IndustryInfo":""
        }
    ],
    "RequestId":"ADDBDECA-E1B2-4988-AE3B-956BF51573EC",
    # 新建业务场景时，您可以根据实际需要导入已有的业务场景列表。
    "BizTypeListImport":[
        {
            "CiteTemplate":false,
            "BizType":"mohongtest2",
            "Source":"custom",
            "IndustryInfo":""
        }
    ]
}
'''
print(response)
```

4.PHP SDK

4.1. 安装

在使用内容检测API PHP SDK前，您需要按照以下步骤安装依赖。

环境准备

支持的PHP版本为5.5以上。

下载SDK

1. 下载PHP SDK。
2. 执行如下命令，安装SDK。

```
composer require alibabacloud/green
```

3. 执行如下命令，引入SDK依赖包。

```
use AlibabaCloud\Green\Green;
```

4.2. 初始化

您需要创建一个Client实例来发起API请求，并根据需要修改Client的配置参数。

背景信息

新建Client时，需要指定服务地域（Region）和阿里云账号的AccessKey信息（包括AccessKey ID和AccessKey Secret）。

- 关于服务地域（Region）的更多信息，请参见接入区域。
- 关于阿里云账号的AccessKey的更多信息，请参见获取AccessKey。

创建用于图片、语音、视频、文本、视频识别的Client

支持的Region包括：

- **cn-shanghai**：华东2（上海）
- **cn-beijing**：华北2（北京）
- **cn-shenzhen**：华南1（深圳）
- **ap-southeast-1**：新加坡

您可以参见以下代码创建Client，用于图片、语音、视频、文本检测：

```
// 设置全局客户端，请替换成您的AccessKey ID、AccessKey Secret。
AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
    ->regionId('cn-shanghai')
    ->asDefaultClient();
```

> **说明** 使用其他Region时，请将代码中所有 `cn-shanghai` 标识替换成对应的Region的标识，如 `cn-beijing` 。

4.3. 内容审核（机审）

4.3.1. 图片审核

本文介绍了如何使用PHP SDK图片审核接口，检测图片中是否包含风险内容。

功能描述

图片审核支持同步检测和异步检测两种方式。

- 同步检测实时返回检测结果。关于参数的详细信息，请参见同步检测。
- 异步检测需要您轮询结果或者通过callback回调通知获取检测结果。关于参数的详细信息，请参见异步检测。

前提条件

已安装PHP依赖。关于安装PHP依赖的具体操作，请参见安装PHP依赖。

> **说明** 请一定按照安装PHP依赖页面中的版本安装，否则会导致调用失败。

（推荐）图片同步检测

接口	描述	支持的地域
ImageSyncScanRequest	提交图片同步检测任务，对图片进行多个风险场景的识别，包括色情、暴恐涉政、广告、二维码、不良场景、Logo（商标台标）识别。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $task1 = array('dataId' => '业务数据ID',
        'url' => '待检测图片链接地址'
    );
    /* 设置待检测的图片，一张图片对应一个检测任务。
     * 多张图片同时检测时，处理时间由最后一张处理完的图片决定。
     * 通常情况下批量检测的平均响应时间比单张检测要长。一次批量提交的图片数越多，响应时间被拉长的概率越高。
     * 代码中以单张图片检测作为示例，如果需要批量检测多张图片，请自行构建多个检测任务。
     * OCR检测按照实际检测的图片张数*检测的卡证类型单价计费。
     */
    $response = Green::v20180509()->imageSyncScan()
        ->body(json_encode(array('tasks' => array($task1), 'scenes' => array('porn'), 'bizType' => '业务场景')))
        ->request();
    print_r($response->toArray());
    if (200 == $response->code) {
        $taskResults = $response->data;
        foreach ($taskResults as $taskResult) {
            if (200 == $taskResult->code) {
                $sceneResults = $taskResult->results;
                foreach ($sceneResults as $sceneResult) {
                    $scene = $sceneResult->scene;
                    $suggestion = $sceneResult->suggestion;
                    // 根据scene和suggetion做相关的处理。
                    print_r($scene);
                    print_r($suggestion);
                }
            } else {
                print_r("task process fail:" + $response->code);
            }
        }
    } else {
        print_r("detect not success. code:" + $response->code);
    }
} catch (Exception $exception) {
    echo $exception->getMessage() . PHP_EOL;
}
```

提交图片异步检测任务

使用PHP SDK对图片进行风险检测，通过异步请求提交检测任务，结果可以通过提交请求时设置callback回调获取，也可以通过接口轮询获取。

接口	描述	支持的地域
ImageAsyncScanRequest	提交图片异步检测任务，对图片进行多个风险场景的识别，包括色情、暴恐涉政、广告、二维码、不良场景、Logo（商标台标）识别。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $task1 = array('dataId' => '业务数据ID',
        'url' => '待检测图片链接地址'
    );
    // 示例：porn（色情）、terrorism（暴恐）。
    $result = Green::v20180509()->imageAsyncScan()
        ->body(json_encode(array("tasks" => array($task1), "scenes" => array("porn", "terro
rism"), "bizType" => "业务场景")))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

查询异步检测结果

接口	描述	支持的地域
ImageAsyncScanResultsRequest	查询图片异步检测任务的结果。支持同时查询多个检测任务的返回结果。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    //请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $result = Green::v20180509()->imageAsyncScanResults()
        ->body(json_encode(array('图片异步检测任务ID')))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

图片检测结果反馈

如果您认为图片检测结果与您的预期不符，可以通过图片检测结果反馈接口，对检测结果进行纠正（系统会根据您反馈的结果，将图片添加到相似图片的黑名单库或者白名单库）。当您再次提交相似的内容进行检测时，以您反馈的label返回结果。

关于接口的说明，请参见检测结果反馈。

接口	描述	支持的Region
ImageScanFeedbackRequest	提交图片检测结果的反馈，以人工反馈的检测结果纠正算法检测结果。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    // 示例：porn（色情）、terrorism（暴恐）。
    $result = Green::v20180509()->imageScanFeedback()
        ->body(json_encode(array("suggestion" => "block", "scenes" => array("porn", "terror
ism"), "url" => "待检测图片链接地址")))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

4.3.2. 视频审核

本文介绍了如何使用PHP SDK视频审核接口，检测视频中是否包含风险内容。

功能描述

视频审核接口支持同步检测和异步检测两种方式。

- 同步检测只支持传递视频的截帧图片序列。关于参数的详细介绍，请参见同步检测。
- 异步检测（推荐）支持对原始视频或视频的截帧图片序列进行检测。关于参数的详细介绍，请参见异步检测。

前提条件

已安装PHP依赖。关于安装PHP依赖的具体操作，请参见安装PHP依赖。

> **说明** 请一定按照安装PHP依赖页面中的版本安装，否则会导致调用失败。

（推荐）提交视频异步检测任务

接口	描述	支持的地域

接口	描述	支持的地域
VideoAsyncScanRequest	提交视频异步检测任务，对视频进行多个风险场景的识别，包括色情、暴恐涉政、广告 、不良场景、Logo（商标台标）识别。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $task1 = array('dataId' => '业务数据ID',
        'url' => '待检测视频链接地址',
    );
    // scenes：检测场景，支持指定多个场景。
    // callback、seed：用于回调通知，可选参数。
    $result = Green::v20180509()->videoAsyncScan()
        ->body(json_encode(array("tasks" => array($task1),
            "scenes" => array("porn", "terrorism"),
            'callback' => '回调地址',
            'seed' => '随机字符串')))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

查询视频异步检测结果

接口	描述	支持的地域
VideoAsyncScanResultsRequest	查询视频异步检测任务的结果。 说明 该方法需要轮询结果，建议使用callback的方式获取结果。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $result = Green::v20180509()->videoAsyncScanResults()
        ->body(json_encode(array('视频检测任务ID')))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

视频截帧同步检测

接口	描述	支持的地域
VideoSyncScanRequest	提交视频同步检测任务，同步检测视频中的风险内容。 **说明** 同步检测只支持传递视频帧序列，不支持检测视频文件，推荐使用异步检测接口。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $task1 = array('frames' => array(['offset' => '0', 'url' => '视频截帧链接地址1'],
        ['offset' => '1', 'url' => '视频截帧链接地址2'],
        ['offset' => '2', 'url' => '视频截帧链接地址3']));
    // scenes：检测场景，支持指定多个场景。
    $result = Green::v20180509()->videoSyncScan()
        ->body(json_encode(array('tasks' => array($task1),
            'scenes' => array('porn', 'terrorism'),
            'bizType' => '业务场景')))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

视频检测结果反馈

如果您认为视频检测结果与您的预期不符，可以通过视频检测结果反馈接口，对检测结果进行纠正（系统会根据您反馈的结果，将视频截帧添加到相似图片的黑名单库或者白名单库）。当您再次提交相似的内容进行检测时，以您反馈的label返回结果。

关于接口的说明，请参见检测结果反馈。

接口	描述	支持的Region
VideoFeedbackRequest	提交视频检测结果的反馈，以人工反馈的检测结果纠正算法检测结果。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    //请替换成你自己的accessKeyId、accessKeySecret
    AlibabaCloud::accessKeyClient($ak['accessKeyId'], $ak['accessKeySecret'])
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    // scenes：检测场景，支持指定多个场景。
    // suggestion：期望的检测结果；pass：正常；block：违规。
    $task1 = array("taskId" => '视频检测任务ID',
        "dataId" => "业务数据ID",
        "url" => "视频链接地址",
        "frames" => array(array("url" => "视频截帧图片链接地址_1", "offset" => "100"), array("
url" => "视频截帧图片链接地址_2", "offset" => "200")),
        "suggestion" => 'block',
        "scenes" => array("ad", "terrorism"),
        "note" => "备注信息");
    $result = Green::v20180509()->videoFeedback()
        ->body(json_encode($task1))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

4.3.3. 文本反垃圾检测

本文介绍了如何使用PHP SDK文本反垃圾接口，对文本内容进行色情、暴恐、涉政等风险进行识别。

功能描述

文本反垃圾接口目前仅支持同步检测。关于参数的详细说明，请参见文本同步检测。

一次请求可以检测多条文本，也可以检测单条文本。按实际检测的文本条数进行计费，请参见计费方式概述。

前提条件

已安装PHP依赖。关于安装PHP依赖的具体操作，请参见安装PHP依赖。

> **说明** 请一定按照安装PHP依赖页面中的版本安装，否则会导致调用失败。

文本内容检测

文本垃圾检测支持自定义关键词，例如添加一些竞品关键词等。如果被检测的文本中包含您添加的关键词，算法会返回您block。

您可以前往内容安全控制台添加关键词，也可以通过API接口添加关键词。

接口	描述	支持的Region
TextScanRequest	提交文本反垃圾检测任务，检测场景参数请传递antispam（scenes=antispam）。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $task1 = array('dataId' => '检测数据ID',
        'content' => '需要检测的文本内容'
    );
    /**
     * 文本垃圾检测：antispam。
     **/
    $result = Green::v20180509()->textScan()
        ->body(json_encode(array('tasks' => array($task1), 'scenes' => array('antispam'), '
bizType' => '业务场景')))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

4.3.4. 语音反垃圾检测

本文介绍了如何使用PHP SDK语音反垃圾接口，检测实时语音流或语音文件中的垃圾内容。

功能描述

语音流检测和语音文件检测均为异步检测，检测结果需要您以轮询或者回调的方式获取。关于调用请求中的检测场景参数scenes，返回结果中的分类参数label，以及操作建议参数suggestion的说明，请参见语音异步检测。

语音检测按照检测的语音文件、语音流的时间长度进行计费，计费粒度为分钟，每天累计检测总时长进行计量统计，每天检测总时长不足一分钟的按照一分钟进行计费。

前提条件

已安装PHP依赖。关于安装PHP依赖的具体操作，请参见安装PHP依赖。

> **说明** 请一定按照安装PHP依赖页面中的版本安装，否则会导致调用失败。

提交语音异步检测任务

接口	描述	支持的地域
VoiceAsyncScanRequest	异步检测语音流或语音文件中是否包含违规内容。语音流格式支持： • RTMP • HTTP • FLV	**cn-shanghai**：华东2（上海）

示例代码

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $task1 = array('dataId' => '业务数据ID', 'url' => '待检测音频链接地址');
    /**
     * 语音垃圾内容检测： antispam。
     * 如果是语音流检测，则live参数修改为true。
     * callback、seed：用于回调通知，可选参数。
     **/
    $result = Green::v20180509()->voiceAsyncScan()
        ->body(json_encode(array('tasks' => array($task1),
            'scenes' => array('antispam'),
            'bizType' => '业务场景',
            'callback' => '回调地址',
            'seed' => '随机字符串')))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

查询语音异步检测结果

接口	描述	支持的地域
VoiceAsyncScanResultsRequest	查询异步语音检测结果。	`cn-shanghai`：华东2（上海）

示例代码

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $result = Green::v20180509()->voiceAsyncScanResults()
        ->body(json_encode(array('异步音频检测任务ID')))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

4.3.5. 文件审核

本文介绍如何使用PHP SDK文件审核接口，检测文件中的文字和图片信息。

功能描述

文件审核目前只支持异步检测（异步检测不会实时返回检测结果）任务。关于参数的详细说明，请参见提交文件检测任务。

前提条件

已安装PHP依赖。关于安装PHP依赖的具体操作，请参见安装PHP依赖。

> **说明** 请一定按照安装PHP依赖页面中的版本安装，否则会导致调用失败。

提交文件异步检测任务

接口	描述	支持的地域
FileAsyncScanRequest	提交文件异步检测任务，对文件中的文字和图片进行检测。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京）

示例代码

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的ccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $task1 = array('dataId' => '检测数据ID',
        'url' => '待检测文件链接地址'
    );
    /**
     * textScenes：检测内容包含文本时，指定检测场景，取值：antispam。
     * imageScenes：检测内容包含图片时，指定检测场景。
     */
    $result = Green::v20180509()->fileAsyncScan()
        ->body(json_encode(array(
            'bizType' => '业务场景',
            'textScenes' => array('antispam'),
            'imageScenes' => array('porn', 'ad'),
            'tasks' => array($task1)
        )))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

获取文件异步检测任务结果

接口	描述	支持的Region
FileAsyncScanResultsRequest	获取文件异步检测结果。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京）

示例代码

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $result = Green::v20180509()->fileAsyncScanResults()
        ->body(json_encode(array('文件审核任务ID')))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

4.3.6. 网页审核

本文介绍如何使用PHP SDK网页审核接口，检测网页中的文字和图片信息。

功能描述

图片审核支持同步检测和异步检测两种方式。

- 同步检测实时返回检测结果。关于参数的详细信息，请参见网页同步检测。
- 异步检测需要您轮询结果或者通过callback回调通知获取检测结果。关于参数的详细信息，请参见网页异步检测。

前提条件

已安装PHP依赖。关于安装PHP依赖的具体操作，请参见安装PHP依赖。

> **说明** 请一定按照安装PHP依赖页面中的版本安装，否则会导致调用失败。

提交网页同步检测任务

接口	描述	支持的地域
WebpageSyncScanRequest	提交网页同步检测任务，对网页中的文字和图片进行检测。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京）

示例代码

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    /**
     * textScenes：检测内容包含文本时，指定检测场景，取值：antispam。
     * imageScenes：检测内容包含图片时，指定检测场景。
     */
    $result = Green::v20180509()->webpageSyncScan()
        ->body(json_encode(array('textScenes' => array('antispam'),
            'imageScenes' => array('porn'),
            'tasks' => array(array('dataId' => '业务数据ID', 'url' => '待检测网页链接'))
        )))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

提交网页异步检测任务

接口	描述	支持的Region
WebpageAsyncScanRequest	提交网页异步检测任务，对网页中的文字和图片进行检测。	• **cn-shanghai**：华东2（上海） • **cn-beijing**：华北2（北京）

示例代码

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    /**
     * textScenes：检测内容包含文本时，指定检测场景，取值：antispam。
     * imageScenes：检测内容包含图片时，指定检测场景。
     */
    $result = Green::v20180509()->webpageAsyncScan()
        ->body(json_encode(array('textScenes' => array('antispam'),
            'imageScenes' => array('porn'),
            'tasks' => array(array('dataId' => '业务数据ID', 'url' => '待检测网页链接'))
        )))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

获取网页异步检测任务结果

接口	描述	支持的Region
WebpageAsyncScanResultsRequest	获取网页异步检测结果。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京）

示例代码

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $result = Green::v20180509()->webpageAsyncScanResults()
        ->body(json_encode(array('网页异步检测任务ID')))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

4.4. 人工审核

4.4.1. 图片人工审核

本文介绍如何使用PHP SDK图片人工审核接口，进行人工审核。

功能描述

如果您认为图片检测结果与您的预期不符，可以进行人工审核。关于参数的详细信息，请参见图片人工审核。

关于该API的接入地址，请参见接入地址（Endpoint）。

前提条件

已安装PHP依赖。关于安装PHP依赖的具体操作，请参见安装PHP依赖。

> **说明** 请一定按照安装PHP依赖页面中的版本安装，否则会导致调用失败。

提交图片人工审核任务

示例代码

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $task1 = array('dataId' => '业务数据ID', 'url' => '待检测图片链接地址');
    $result = Green::v20180509()->imageAsyncManualScan()
        ->body(json_encode(array('tasks' => array($task1),
            'callback' => '回调地址',
            'seed' => '随机字符串')))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

查询图片人工审核任务结果

示例代码

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $result = Green::v20180509()->imageAsyncManualScanResults()
        ->body(json_encode(array('图片人工审核异步检测任务ID')))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

4.4.2. 视频人工审核

本文介绍如何使用PHP SDK视频人工审核接口，进行人工审核。

功能描述

如果您认为视频检测结果与您的预期不符，可以进行人工审核。关于参数的详细信息，请参见视频人工审核API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装PHP依赖。关于安装PHP依赖的具体操作，请参见安装PHP依赖。

> **说明** 请一定按照安装PHP依赖页面中的版本安装，否则会导致调用失败。

提交视频人工审核任务

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $task1 = array('dataId' => '业务数据ID', 'url' => '待检测视频链接地址');
    $result = Green::v20180509()->videoAsyncManualScan()
        ->body(json_encode(array("tasks" => array($task1),
            "callback" => "回调地址",
            "seed" => "随机字符串")))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

查询视频人工审核任务结果

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $result = Green::v20180509()->videoAsyncManualScanResults()
        ->body(json_encode(array('视频人工检测任务ID')))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

4.4.3. 文本人工审核

本文介绍如何使用PHP SDK文本人工审核接口，进行人工审核。

功能描述

如果您认为文本检测结果与您的预期不符，可以进行人工审核。关于参数的详细信息，请参见文本人工审核API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装PHP依赖。关于安装PHP依赖的具体操作，请参见安装PHP依赖。

> **说明** 请一定按照安装PHP依赖页面中的版本安装，否则会导致调用失败。

提交文本人工审核任务

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $task1 = array('dataId' => '业务数据ID', 'content' => '待检测文本');
    //callback、seed：用于回调通知，可选参数。
    $response = Green::v20180509()->textAsyncManualScan()
        ->body(json_encode(array('tasks' => array($task1),
            'callback' => '回调地址',
            'seed' => '随机字符串')))
        ->request();
    print_r($response->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

查询文本人工审核任务结果

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $response = Green::v20180509()->textAsyncManualScanResults()
        ->body(json_encode(array('文本人工审核任务ID')))
        ->request();
    print_r($response->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

4.4.4. 语音人工审核

本文介绍如何使用PHP SDK语音人工审核接口，进行人工审核。

功能描述

如果您认为语音检测结果与您的预期不符，可以进行人工审核。关于参数的详细信息，请参见语音人工审核API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装PHP依赖。关于安装PHP依赖的具体操作，请参见安装PHP依赖。

> **说明** 请一定按照安装PHP依赖页面中的版本安装，否则会导致调用失败。

提交语音人工审核任务

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $task1 = array('dataId' => '业务数据ID',
        'url' => '待检测音频链接地址',
    );
    // callback、seed：用于回调通知，可选参数。
    $result = Green::v20180509()->voiceAsyncManualScan()
        ->body(json_encode(array('tasks' => array($task1),
            'callback' => '回调地址', 'seed' => '随机字符串')))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

查询语音人工审核任务结果

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的ccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $result = Green::v20180509()->voiceAsyncManualScanResults()
        ->body(json_encode(array('音频人工审核任务ID')))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

4.5. 图片OCR识别

本文介绍了如何使用PHP SDK图片OCR接口，识别图片中的文字或卡证信息。

功能描述

通用OCR除了能够识别普通图片中的文字，还能识别结构化的卡证上的文字。关于参数的详细说明，请参见图片OCR检测API文档。

前提条件

已安装PHP依赖。关于安装PHP依赖的具体操作，请参见安装PHP依赖。

> **说明** 请一定按照安装PHP依赖页面中的版本安装，否则会导致调用失败。

提交图片同步检测任务

接口	描述	支持的Region
ImageSyncScanRequest	提交图片OCR同步识别任务，对图片中的文字进行识别（scene=ocr）。	• *cn-shanghai* • *cn-beijing* • *cn-shenzhen* • *ap-southeast-1*

示例代码

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $task1 = array('dataId' => '业务数据ID',
        'url' => '待检测图片链接地址'
    );
    // 示例：身份证正面识别。
    $extras = array('card' => 'id-card-front');
    /* 设置待检测的图片，一张图片对应一个检测任务。
     * 多张图片同时检测时，处理时间由最后一张处理完的图片决定。
     * 通常情况下批量检测的平均响应时间比单张检测要长。一次批量提交的图片数越多，响应时间被拉长的概率越高。
     * 代码中以单张图片检测作为示例，如果需要批量检测多张图片，请自行构建多个检测任务。
     * OCR检测按照实际检测的图片张数*检测的卡证类型单价计费。
     */
    $result = Green::v20180509()->imageSyncScan()
        ->body(json_encode(array('tasks' => array($task1), 'scenes' => array('ocr'), 'extras' => array($extras))))
        ->request();
    print_r($result->toArray());
} catch (Exception $exception) {
    echo $exception->getMessage() . PHP_EOL;
}
```

4.6. 人脸识别

4.6.1. 人脸属性检测

本文介绍了如何使用PHP SDK人脸属性检测接口，对指定的人脸图片（公网图片和本地图片）进行属性检测。人脸属性检测仅支持以图片作为检测媒介。

功能描述

人脸属性检测能够识别图片中的人脸属性信息，包括人脸模糊度、人脸角度、人脸位置、微笑程度、是否戴眼镜、是否戴口罩、是否戴帽子、是否有胡子、是否有刘海、头发类型等。关于参数的详细说明，请参见人脸属性检测API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装PHP依赖。关于安装PHP依赖的具体操作，请参见安装PHP依赖。

> **说明** 请一定按照安装PHP依赖页面中的版本安装，否则会导致调用失败。

提交人脸属性检测任务

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $result = Green::v20180509()->detectFace()
        ->body(json_encode(array('dataId' => '业务数据ID',
            'url' => '待检测图片链接地址',
            'bizType' => '业务场景')))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

4.6.2. 自定义人脸检索

4.6.2.1. 自定义人脸检索

本文介绍了如何使用PHP SDK自定义人脸检索接口，对指定的人脸图片进行自定义人脸检索。

功能描述

自定义人脸检索根据您输入的待识别人脸图片（face），在指定人脸库（group）中查找并返回最相似的Top 5个体（person），返回的Top 5个体按照相似度从大到小排序。

检索人脸时，必须指定要检索的分组信息（最多指定一个分组）。关于参数的详细说明，请参见自定义人脸检索API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装PHP依赖。关于安装PHP依赖的具体操作，请参见安装PHP依赖。

> **说明** 请一定按照安装PHP依赖页面中的版本安装，否则会导致调用失败。

提交人脸图片检索任务

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    // url：待检索图片url，必选；groupId：检索库的Id，必选。
    $task1 = array('dataId' => '业务数据ID',
        'url' => '待检测图片链接地址',
        'extras' => array('groupId' => '个体组ID')
    );
    // scenes：检测场景，取值：sface-n，表示自定义人脸检索。
    $result = Green::v20180509()->imageSyncScan()
        ->body(json_encode(array("tasks" => array($task1),
            "scenes" => array("sface-n"))))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

4.6.2.2. 新建个体

本文介绍了如何使用PHP SDK新建个体信息。

功能描述

新建个体时，必须指定个体要加入的分组。关于参数的详细说明，请参见新建个体API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装PHP依赖。关于安装PHP依赖的具体操作，请参见安装PHP依赖。

> **说明** 请一定按照安装PHP依赖页面中的版本安装，否则会导致调用失败。

新建个体任务

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    /**
     * personId: 用户自定义个体ID，必填。
     * groupIds: 用户自定义组ID列表，必填。
     * name: 用户名称，非必填。
     * note: 备注信息，非必填。
     */
    $person = array('personId' => '个体ID',
        'groupIds' => array('个体组ID_0', '个体组ID_1'),
        'name' => '用户名称',
        'note' => '备注信息'
    );
    $result = Green::v20180509()->addPerson()
        ->body(json_encode($person))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

4.6.2.3. 设置个体

本文介绍了如何使用PHP SDK设置个体信息。

功能描述

设置个体信息用于给个体添加备注信息，属于可选步骤。如果您设置了个体信息，则在自定义检索的返回结果中会包含该信息。关于参数的详细说明，请参见设置个体API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装PHP依赖。关于安装PHP依赖的具体操作，请参见安装PHP依赖。

> **说明** 请一定按照安装PHP依赖页面中的版本安装，否则会导致调用失败。

提交设置个体任务

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    /**
     * personId: 用户自定义个体ID，必填。
     * name: 用户名称，非必填。
     * note: 备注信息，非必填。
     */
    $person = array('personId' => '个体ID',
        'name' => '用户名称',
        'note' => '备注信息'
    );
    $result = Green::v20180509()->setPerson()
        ->body(json_encode($person))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

4.6.2.4. 查询个体信息

本文介绍了如何使用PHP SDK查询个体信息。

功能描述

关于参数的详细说明，请参见查询个体信息API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装PHP依赖。关于安装PHP依赖的具体操作，请参见安装PHP依赖。

> **说明** 请一定按照安装PHP依赖页面中的版本安装，否则会导致调用失败。

查询个体信息任务

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    // personId: 用户自定义个体ID，必填。
    $person = array('personId' => '个体ID');
    $result = Green::v20180509()->getPerson()
        ->body(json_encode($person))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

4.6.2.5. 查询人脸信息

本文介绍了如何使用PHP SDK查询人脸信息。

功能描述

根据个体ID和人脸ID查询当前个体的人脸ID以及人脸图片路径等信息。关于查询人脸的具体参数说明，请参见查询人脸图片API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装PHP依赖。关于安装PHP依赖的具体操作，请参见安装PHP依赖。

> **说明** 请一定按照安装PHP依赖页面中的版本安装，否则会导致调用失败。

查询人脸信息任务

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    // personId：已新建的personId，必选。faceIds：已新增的人脸ID，必选。
    $person = array('personId' => '个体ID',
        'faceIds' => array('人脸ID_1')
    );
    $result = Green::v20180509()->getFaces()
        ->body(json_encode($person))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

4.6.2.6. 删除个体

本文介绍了如何使用PHP SDK删除个体信息。

功能描述

删除个体时，个体对应的图片以及信息均会被删除。关于参数的详细说明，请参见删除个体API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装PHP依赖。关于安装PHP依赖的具体操作，请参见安装PHP依赖。

> **说明** 请一定按照安装PHP依赖页面中的版本安装，否则会导致调用失败。

删除个体任务

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    // personId：已新建的personId，必选。
    $person = array('personId' => '个体ID'
    );
    $result = Green::v20180509()->deletePerson()
        ->body(json_encode($person))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

4.6.2.7. 新增人脸

本文介绍了如何使用PHP SDK新增个体的人脸图片。

功能描述

为个体新增一组人脸图片，一个个体最多允许包含20张人脸图片。关于参数的说明，请参见新增人脸API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装PHP依赖。关于安装PHP依赖的具体操作，请参见安装PHP依赖。

> **说明** 请一定按照安装PHP依赖页面中的版本安装，否则会导致调用失败。

新增人脸图片任务

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    // personId：已新建的personId，必选；urls：人脸图片url，必选。图片只能包含一个人脸。
    $person = array('personId' => '个体ID',
        'urls' => array('人脸图片链接地址_1')
    );
    $result = Green::v20180509()->addFaces()
        ->body(json_encode($person))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

4.6.2.8. 删除人脸

本文介绍了如何使用PHP SDK，删除无用的个体人脸图片。

功能描述

删除人脸图片时，必须指定要删除的图片ID以及对应的个体ID信息。关于参数的详细说明，请参见删除人脸API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装PHP依赖。关于安装PHP依赖的具体操作，请参见安装PHP依赖。

> **说明** 请一定按照安装PHP依赖页面中的版本安装，否则会导致调用失败。

提交删除人脸任务

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    // personId：已新建的personId，必选；faceIds：已新增的人脸ID，必选。
    $person = array('personId' => '个体ID',
        'faceIds' => array('人脸ID_0')
    );
    $result = Green::v20180509()->deleteFaces()
        ->body(json_encode($person))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

4.6.2.9. 查询组列表

本文介绍了如何使用PHP SDK查询所有人脸分组ID。

功能描述

查询所有的个体组ID。关于参数的详细说明，请参见查询组列表API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装PHP依赖。关于安装PHP依赖的具体操作，请参见安装PHP依赖。

> **说明** 请一定按照安装PHP依赖页面中的版本安装，否则会导致调用失败。

查询所有分组ID任务

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $result = Green::v20180509()->getGroups()
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

4.6.2.10. 查询个体列表

本文介绍了如何使用PHP SDK查询个体列表信息。

功能描述

每个个体必须属于一个分组，在调用该接口时指定某个分组，则返回该分组下的所有个体ID列表。关于参数的详细说明，请参见查询个体列表API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装PHP依赖。关于安装PHP依赖的具体操作，请参见安装PHP依赖。

> **说明** 请一定按照安装PHP依赖页面中的版本安装，否则会导致调用失败。

查询个体列表任务

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    // groupId：待查询的组ID。
    $groupId = array('groupId' => '个体组ID');
    $result = Green::v20180509()->getPersons()
        ->body(json_encode($groupId))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

4.6.2.11. 添加个体到分组

本文介绍了如何使用PHP SDK添加个体到指定分组。

功能描述

将一个个体添加到一个或者多个个体组。关于参数的详细说明，请参见添加个体到分组API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装PHP依赖。关于安装PHP依赖的具体操作，请参见安装PHP依赖。

> **说明** 请一定按照安装PHP依赖页面中的版本安装，否则会导致调用失败。

添加个体到分组任务

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    // personId：已添加的personId，必选；groupIds：用户自定义组Id，类型为数组，可以指定多个组，必选
。
    $person = array('personId' => '个体ID',
        'groupIds' => array('个体组Id_1')
    );
    $result = Green::v20180509()->addGroups()
        ->body(json_encode($person))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

4.6.2.12. 删除分组中个体

本文介绍了如何使用PHP SDK从指定分组中删除个体信息。

功能描述

删除分组中个体不会删除个体对应的信息以及图片，只是解除个体与该分组的绑定关系。关于参数的详细说明，请参见删除分组中的个体API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装PHP依赖。关于安装PHP依赖的具体操作，请参见安装PHP依赖。

> **说明** 请一定按照安装PHP依赖页面中的版本安装，否则会导致调用失败。

从指定分组删除个体任务

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    // personId：已新建的personId，必选；groupIds：用户自定义组ID列表，必选。
    $person = array('personId' => '个体ID',
        'groupIds' => array('个体组ID_1')
    );
    $result = Green::v20180509()->deleteGroups()
        ->body(json_encode($person))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

4.7. 其他

4.7.1. 相似图检索

4.7.1.1. 创建相似图样本库

本文介绍了如何使用PHP SDK，创建相似图样本库。

功能描述

通过该接口创建相似图样本图库。关于参数的详细说明，请参见创建图库API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装PHP依赖。关于安装PHP依赖的具体操作，请参见安装PHP依赖。

> **说明** 请一定按照安装PHP依赖页面中的版本安装，否则会导致调用失败。

创建相似图样本任务

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $result = Green::v20180509()->addSimilarityLibrary()
        ->body(json_encode(array('name' => '相似图库名称')))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

4.7.1.2. 相似图检索

本文介绍了如何使用PHP SDK，对指定的图片进行相似图检索。

功能描述

相似图检索帮助您从图库中检索出与给定的图片相似的若干张图片。关于参数的详细说明，请参见相似图检索API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装PHP依赖。关于安装PHP依赖的具体操作，请参见安装PHP依赖。

> **说明** 请一定按照安装PHP依赖页面中的版本安装，否则会导致调用失败。

提交相似图检索任务

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $task1 = array('dataId' => '业务数据ID',
        'url' => '待检测图片链接地址'
    );
    /**
     * 设置要检测的场景，计费是按照该处传递的场景进行。
     * similarity：表示进行相似图检索。
     */
    $result = Green::v20180509()->imageSyncScan()
        ->body(json_encode(array("tasks" => array($task1), "scenes" => array("similarity"))
))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

4.7.1.3. 增加相似图样本

本文介绍了如何使用PHP SDK，将指定的相似图图片添加到对应图库。

功能描述

添加相似图样本时，您可以为图片设置标签。如果样本图片在检索时被命中，其标签信息也会被返回。关于参数的详细说明，请参见增加样本图片API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装PHP依赖。关于安装PHP依赖的具体操作，请参见安装PHP依赖。

> **说明** 请一定按照安装PHP依赖页面中的版本安装，否则会导致调用失败。

增加相似图样本任务

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $task1 = array('dataId' => uniqid(),
        'url' => '样本图片链接地址',
        'tag' => ["自定义标签1", "自定义标签2", "自定义标签3"]
    );
    $result = Green::v20180509()->addSimilarityImage()
        ->body(json_encode(array("tasks" => array($task1), "library" => "样本图库名称")))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

4.7.1.4. 查询相似图库列表

本文介绍了如何使用PHP SDK，分页查询相似图库及其元素信息。

功能描述

调用该接口分页查询所有相似图库及其元素信息。关于参数的详细说明，请参见查询相似图库列表API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装PHP依赖。关于安装PHP依赖的具体操作，请参见安装PHP依赖。

> **说明** 请一定按照安装PHP依赖页面中的版本安装，否则会导致调用失败。

查询相似图库列表任务

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $result = Green::v20180509()->listSimilarityLibraries()
        ->body(json_encode(array("pageSize" => "5", "currentPage" => "1")))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

4.7.1.5. 查询指定相似图库信息

本文介绍了如何使用PHP SDK，查询指定相似图库信息。

功能描述

调用该接口查询指定相似图库信息。关于参数的详细说明，请参见查询指定相似图库API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装PHP依赖。关于安装PHP依赖的具体操作，请参见安装PHP依赖。

> **说明** 请一定按照安装PHP依赖页面中的版本安装，否则会导致调用失败。

查询指定相似图库任务

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $result = Green::v20180509()->getSimilarityLibrary()
        ->body(json_encode(array("name" => "样本图库名称")))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

4.7.1.6. 查询相似图样本图片列表

本文介绍了如何使用PHP SDK，分页查询指定相似图库下的所有样本图片信息。

功能描述

调用该接口分页查询指定相似图库下的所有样本图片信息。关于参数的详细说明，请参见查询样本图片列表API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装PHP依赖。关于安装PHP依赖的具体操作，请参见安装PHP依赖。

> **说明** 请一定按照安装PHP依赖页面中的版本安装，否则会导致调用失败。

查询相似图样本图片列表任务

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $result = Green::v20180509()->listSimilarityImages()
        ->body(json_encode(array("library" => "样本图库名称", "pageSize" => "5", "currentPage
" => "1")))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

4.7.1.7. 查询相似图样本图片详情

本文介绍了如何使用PHP SDK，查询相似图样本图片详情。

功能描述

调用该接口查询相似图样本图片详情。关于参数的详细说明，请参见查询样本图库详情API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装PHP依赖。关于安装PHP依赖的具体操作，请参见安装PHP依赖。

> **说明** 请一定按照安装PHP依赖页面中的版本安装，否则会导致调用失败。

查询相似图样本图片详情任务

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $result = Green::v20180509()->getSimilarityImage()
        ->body(json_encode(array("library" => "样本图库名称", "dataId" => "图片样本ID")))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

4.7.1.8. 移除相似图样本

本文介绍了如何使用PHP SDK，将相似图样本从图库中移除。

功能描述

移除操作在1分钟之内生效。一次最多允许移除100张样本图片。关于参数的详细说明，请参见删除样本图片API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装PHP依赖。关于安装PHP依赖的具体操作，请参见安装PHP依赖。

> **说明** 请一定按照安装PHP依赖页面中的版本安装，否则会导致调用失败。

移除相似图样本任务

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $result = Green::v20180509()->deleteSimilarityImage()
        ->body(json_encode(array('dataIds' => array('样本图片id_1'))))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

4.7.1.9. 删除相似图库

本文介绍了如何使用PHP SDK，删除指定相似图库。

功能描述

只有当相似图库中无图片样本时，才允许被删除。关于参数的详细说明，请参见删除图库API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装PHP依赖。关于安装PHP依赖的具体操作，请参见安装PHP依赖。

> **说明** 请一定按照安装PHP依赖页面中的版本安装，否则会导致调用失败。

删除相似图库任务

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $result = Green::v20180509()->deleteSimilarityLibrary()
        ->body(json_encode(array('name' => '样本图库名称')))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

4.7.2. 自定义文本库

本文介绍了如何使用PHP SDK管理自定义文本库，以满足文本反垃圾检测场景的个性化需求。

功能描述

根据文本类型的不同，文本库分为关键词文本库和相似文本库；根据管控目的不同，文本库分为白名单、黑名单、疑似名单。关于参数的详细信息，请参见自定义文本库API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装PHP依赖。关于安装PHP依赖的具体操作，请参见安装PHP依赖。

> **说明** 请一定按照安装PHP依赖页面中的版本安装，否则会导致调用失败。

查询文本库列表

- 查询关键词文本库列表

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $response = Green::v20170823()->describeKeywordLib()
        ->withServiceModule('open_api')
        ->request();
    print_r($response);
    $allLibs = $response->KeywordLibList;
    $textAntispamKeywordLibs = array();
    foreach ($allLibs as $keywordLib) {
        $libType = $keywordLib->LibType;
        $resourceType = $keywordLib->ResourceType;
        $source = $keywordLib->Source;
        # 查询自定义的关键词文本库。
        if ("textKeyword" == $libType and "TEXT" == $resourceType and "MANUAL" == $source
) {
            array_push($textAntispamKeywordLibs, $keywordLib);
        }
        # 查询回流的关键词文本库。
        if ("textKeyword" == $libType and "TEXT" == $resourceType and "FEEDBACK" == $sour
ce) {
            array_push($textAntispamKeywordLibs, $keywordLib);
        }
    }
    print_r($textAntispamKeywordLibs);
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

- 查询相似文本库列表（包含自定义和系统回流的相似文本库）

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的ccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $response = Green::v20170823()->describeKeywordLib()
        ->withServiceModule('open_api')
        ->request();
    print_r($response->toArray());
    $allLibs = $response->KeywordLibList;
    $similarTextLibs = array();
    foreach ($allLibs as $keywordLib) {
        $libType = $keywordLib->LibType;
        $resourceType = $keywordLib->ResourceType;
        $source = $keywordLib->Source;
        # 获取文本反垃圾自定义的关键词文本库。
        if ('similarText' == $libType and 'TEXT' == $resourceType and 'MANUAL' == $source
) {
            array_push($similarTextLibs, $keywordLib);
        }
        # 获取文本反垃圾回流的关键词文本库。
        if ('similarText' == $libType and 'TEXT' == $resourceType and 'FEEDBACK' == $sour
ce) {
            array_push($similarTextLibs, $keywordLib);
        }
    }
    print_r($similarTextLibs);
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

创建文本库

- 创建关键词文本库

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $response = Green::v20170823()->createKeywordLib()
        ->withServiceModule('open_api')
        ->withName('自定义的关键词文本库')
        ->withResourceType('TEXT')
        ->withLibType('textKeyword')
        ->withCategory('BLACK')
        ->request();
    print_r($response->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

- 创建相似文本库

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的ccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $response = Green::v20170823()->createKeywordLib()
        ->withServiceModule('open_api')
        ->withName('测试相似文本库')
        ->withResourceType('TEXT')
        ->withLibType('similarText')
        ->withCategory('BLACK')
        ->request();
    print_r($response->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

修改文本库

更新文本库库名以及修改文本库所适用的业务场景（BizType）。

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $response = Green::v20170823()->updateKeywordLib()
        ->withId('文本库ID')
        ->withName('文本库修改名称')
        ->request();
    print_r($response->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

删除文本库

> **注意** 删除文本库也将删除文本库下的文本。系统回流的文本库不允许删除。

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $response = Green::v20170823()->deleteKeywordLib()
        ->withId('文本库ID')
        ->request();
    print_r($response->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

查找文本

默认分页查询所有文本。如果设置了Keyword字段，将模糊查找包含该字段值的文本。

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的ccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $response = Green::v20170823()->describeKeyword()
        ->withKeywordLibId('文本库ID')
        ->request();
    print_r($response->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

添加文本

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $response = Green::v20170823()->createKeyword()
        ->withKeywordLibId()
        ->withKeywords()
        ->request();
    print_r($response->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

删除文本

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $response = Green::v20170823()->deleteKeyword()
        ->withKeywordLibId('文本库ID')
        ->withIds('[\'文本ID_1\',\'文本ID_2\']')
        ->request();
    print_r($response->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

4.7.3. 自定义图库

本文介绍了如何使用PHP SDK管理自定义图库。

功能描述

您可以自定义智能鉴黄、暴恐涉政识别、图片或视频广告的图片样本，满足个性化内容管控需求。关于参数的详细信息，请参见创建图库API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装PHP依赖。关于安装PHP依赖的具体操作，请参见安装PHP依赖。

> **说明** 请一定按照安装PHP依赖页面中的版本安装，否则会导致调用失败。

查询自定义图库列表

您可以使用以下代码查询用户图库列表（包括用户自定义的图库列表和系统回流图库）：

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $result = Green::v20170823()->describeImageLib()
        ->withServiceModule('open_api')
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

创建自定义图库

您可以使用以下代码创建自定义图库：

> **说明** 请根据您所需要的业务场景设置不同的参数。

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $result = Green::v20170823()->createImageLib()
        ->withServiceModule('open_api')
        ->withName('图库名称')
        ->withScene('图库的使用场景')
        ->withCategory('图库类型')
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

修改自定义图库

您可以使用以下代码修改自定义图库的名称及其适用的业务场景（BizType）：

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $result = Green::v20170823()->updateImageLib()
        ->withId('图片ID')
        ->withName('图库名称')
        ->withScene('图库的使用场景')
        ->withCategory('图库类型')
        ->withBizTypes(array('业务场景1', '业务场景2'))
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

删除自定义图库

您可以使用以下代码删除自定义图库：

> **注意** 删除自定义图库时，图库下的所有图片也将被删除。

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $result = Green::v20170823()->deleteImageLib()
        ->withId('图库ID')
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

查询自定义图库图片列表

您可以使用以下代码查询自定义图库中所有已添加的图片列表：

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的ccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    $result = Green::v20170823()->describeImageFromLib()
        ->withImageLibId('图库Id')
        ->withPageSize('10')
        ->withCurrentPage('1')
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

删除自定义图片

您可以使用以下代码删除自定义图库中的多张自定义图片：

```
<?php
use AlibabaCloud\Client\AlibabaCloud;
use AlibabaCloud\Client\Exception\ClientException;
use AlibabaCloud\Client\Exception\ServerException;
use AlibabaCloud\Green\Green;
try {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    AlibabaCloud::accessKeyClient('您的AccessKey ID', '您的AccessKey Secret')
        ->regionId('cn-shanghai')
        ->asDefaultClient();
    // 多张图片请使用英文逗号隔开。
    $result = Green::v20170823()->deleteImageFromLib()
        ->withIds("['图片ID_1','图片ID_2']")
        ->request();
    print_r($result->toArray());
} catch (ClientException $exception) {
    echo $exception->getMessage() . PHP_EOL;
} catch (ServerException $exception) {
    echo $exception->getMessage() . PHP_EOL;
    echo $exception->getErrorCode() . PHP_EOL;
    echo $exception->getRequestId() . PHP_EOL;
    echo $exception->getErrorMessage() . PHP_EOL;
}
```

5.Go SDK

5.1. 安装

在使用内容检测API Go SDK前，您需要按照以下步骤安装依赖。

环境准备

支持的Go版本为1.10.0以上。

安装SDK

1. 下载Go SDK。
2. 执行如下命令，安装SDK。

```
go get -u github.com/aliyun/alibaba-cloud-sdk-go/sdk
```

3. 执行如下命令，引入SDK依赖包。

```
import "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
```

5.2. 初始化

您需要创建一个客户端（Client）实例来发起API请求，并根据需要修改实例的配置参数。

背景信息

新建实例时，需要指定服务地域（Region）和阿里云账号的AccessKey信息（包括AccessKey ID和AccessKey Secret）。

- 关于服务地域的更多信息，请参见接入区域。
- 关于阿里云账号AccessKey的更多信息，请参见获取AccessKey。

创建用于图片、语音、视频、文本、视频识别的客户端实例

支持的地域包括：

- **cn-shanghai**：华东2（上海）
- **cn-beijing**：华北2（北京）
- **cn-shenzhen**：华南1（深圳）
- **ap-southeast-1**：新加坡

您可以参见以下代码创建实例，用于图片、语音、视频、文本检测：

```
// 请替换成您的AccessKey ID、AccessKey Secret。
client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的AccessKey Secret")
if err != nil {
   // handle exceptions
   panic(err)
}
```

> **说明** 使用其他地域时，请将代码中所有 `cn-shanghai` 标识替换成对应的Region的标识，如 `cn-beijing` 。

5.3. 内容审核（机审）

5.3.1. 图片审核

本文介绍了如何使用Go SDK图片审核接口，检测图片中是否包含风险内容。

功能描述

图片审核支持同步检测和异步检测两种方式。

- 同步检测实时返回检测结果。关于参数的详细信息，请参见同步检测。
- 异步检测需要您轮询结果或者通过callback回调通知获取检测结果。关于参数的详细信息，请参见异步检测。

支持传入检测的图片内容媒介包括互联网图片URL、本地图片文件路径和二进制图片文件流。

前提条件

已安装Go依赖。关于安装Go依赖的具体操作，请参见安装Go依赖。

> **说明** 请一定按照安装Go依赖页面中的版本安装，否则会导致调用失败。

（推荐）图片同步检测

接口	描述	支持的地域
ImageSyncScanRequest	提交图片同步检测任务，对图片进行多个风险场景的识别，包括色情、暴恐涉政、广告、二维码、不良场景、Logo（商标台标）识别。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acces
sKey Secret")
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   task1 := map[string]interface{}{"dataId": "检测数据ID", "url": "待检测图片链接地址"}
   // scenes：检测场景，支持指定多个场景。
   content, _ := json.Marshal(
      map[string]interface{}{
         "tasks": task1, "scenes": [...]string{"porn", "terrorism"}, "bizType": "业务场景",
      },
   )
   request := green.CreateImageSyncScanRequest()
   request.SetContent(content)
   response, _err := client.ImageSyncScan(request)
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

提交图片异步检测任务

使用Go SDK对图片进行风险检测，通过异步请求提交检测任务，结果可以通过提交请求时设置callback回调获取，也可以通过接口轮询获取。

异步检测支持的输入内容与同步检测一致（本地文件、图片URL、图片二进制流），以下仅以图片URL作为输入示例。

接口	描述	支持的地域
ImageAsyncScanRequest	提交图片异步检测任务，对图片进行多个风险场景的识别，包括色情、暴恐涉政、广告、二维码、不良场景、Logo（商标台标）识别。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acces
sKey Secret")
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   task1 := map[string]interface{}{"dataId": "检测数据ID", "url": "待检测图片链接地址"}
   // scenes：检测场景，支持指定多个场景。
   content, _ := json.Marshal(
      map[string]interface{}{
         "tasks": [...]map[string]interface{}{task1}, "scenes": [...]string{"porn", "terror
ism"},
         "bizType": "业务场景",
      },
   )
   request := green.CreateImageAsyncScanRequest()
   request.SetContent(content)
   response, _err := client.ImageAsyncScan(request)
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

查询异步检测结果

接口	描述	支持的地域
ImageAsyncScanResultsRequest	查询图片异步检测任务的结果。支持同时查询多个检测任务的返回结果。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acces
sKey Secret")
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   content, _ := json.Marshal(
      [...]string{"图片异步检测任务ID"},
   )
   request := green.CreateImageAsyncScanResultsRequest()
   request.SetContent(content)
   response, _err := client.ImageAsyncScanResults(request)
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

图片检测结果反馈

如果您认为图片检测结果与您的预期不符，可以通过图片检测结果反馈接口，对检测结果进行纠正（系统会根据您反馈的结果，将图片添加到相似图片的黑名单库或者白名单库）。当您再次提交相似的内容进行检测时，以您反馈的label返回结果。

关于接口的说明，请参见检测结果反馈。

接口	描述	支持的Region
ImageScanFeedbackRequest	提交图片检测结果的反馈，以人工反馈的检测结果纠正算法检测结果。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, _err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acce
ssKey Secret")
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
  // scenes：检测场景，支持指定多个场景。
   // suggestion：期望的检测结果；pass：正常；block：违规。
   content, _ := json.Marshal(
      map[string]interface{}{
         "suggestion": "block", "scenes": [...]string{"porn", "terrorism"}, "url": "待检测图
片链接地址",
      },
   )
   request := green.CreateImageScanFeedbackRequest()
   request.SetContent(content)
   response, err := client.ImageScanFeedback(request)
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

5.3.2. 视频审核

本文介绍了如何使用Go SDK视频审核接口，检测视频中是否包含风险内容。

功能描述

视频审核接口支持同步检测和异步检测两种方式。

- 同步检测只支持传递视频的截帧图片序列。关于参数的详细介绍，请参见同步检测。
- 异步检测（推荐）支持对原始视频或视频的截帧图片序列进行检测。关于参数的详细介绍，请参见异步检测。

前提条件

已安装Go依赖。关于安装Go依赖的具体操作，请参见安装Go依赖。

> **说明** 请一定按照安装Go依赖页面中的版本安装，否则会导致调用失败。

（推荐）提交视频异步检测任务

接口	描述	支持的地域
VideoAsyncScanRequest	提交视频异步检测任务，对视频进行多个风险场景的识别，包括色情、暴恐涉政、广告 、不良场景、Logo（商标台标）识别。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

- 提交视频URL进行检测

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acc
essKey Secret")
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   task := map[string]interface{}{"dataId": "检测数据ID", "url": "待检测视频链接地址"}
  // scenes：检测场景，支持指定多个场景。
   // callback、seed用于回调通知，可选参数。
   content, _ := json.Marshal(
      map[string]interface{}{
         "tasks": [...]map[string]interface{}{task}, "scenes": [...]string{"porn", "terro
rism"},
         "bizType": "业务场景", "callback": "回调地址", "seed": "随机字符串",
      },
   )
   request := green.CreateVideoAsyncScanRequest()
   request.SetContent(content)
   response, _err := client.VideoAsyncScan(request)
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()
))
   }
   fmt.Println(response.GetHttpContentString())
}
```

- 提交视频直播流进行检测

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acc
essKey Secret")
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   // url填写直播流地址。
   task := map[string]interface{}{"dataId": "检测数据ID", "url": "待检测视频链接地址"}
   // scenes：检测场景，支持指定多个场景。
   // callback、seed用于回调通知，可选参数。
   content, _ := json.Marshal(
      map[string]interface{}{
         "tasks": [...]map[string]interface{}{task}, "scenes": [...]string{"porn", "terro
rism"},
         "live": "true", "bizType": "业务场景", "callback": "回调地址", "seed": "随机字符串"
,
      },
   )
   request := green.CreateVideoAsyncScanRequest()
   request.SetContent(content)
   response, _err := client.VideoAsyncScan(request)
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()
))
   }
   fmt.Println(response.GetHttpContentString())
}
```

- 提交视频语音进行综合检测

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acc
essKey Secret")
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   // url填写直播流地址。
   task := map[string]interface{}{"dataId": "检测数据ID", "url": "待检测视频链接地址"}
   // scenes：检测场景，支持指定多个场景。
   // callback、seed用于回调通知，可选参数。
   content, _ := json.Marshal(
      map[string]interface{}{
         "tasks": [...]map[string]interface{}{task}, "scenes": [...]string{"porn", "terro
rism"},
         "live": "true", "audioScenes": [...]string{"antispam"}, "bizType": "业务场景",
         "callback": "回调地址",
         "seed": "随机字符串",
      },
   )
   request := green.CreateVideoAsyncScanRequest()
   request.SetContent(content)
   response, _err := client.VideoAsyncScan(request)
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()
))
   }
   fmt.Println(response.GetHttpContentString())
}
```

查询视频异步检测结果

接口	描述	支持的地域
VideoAsyncScanResultsRequest	查询视频异步检测任务的结果。 说明 该方法需要轮询结果，建议使用callback的方式获取结果。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acces
sKey Secret")
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   content, _ := json.Marshal(
      [...]string{"视频异步检测任务taskId"},
   )
   request := green.CreateVideoAsyncScanResultsRequest()
   request.SetContent(content)
   response, _err := client.VideoAsyncScanResults(request)
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

视频截帧同步检测

接口	描述	支持的地域
VideoSyncScanRequest	提交视频同步检测任务，同步检测视频中的风险内容。 **说明** 同步检测只支持传递视频帧序列，不支持检测视频文件，推荐使用异步检测接口。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

以下示例代码中使用帧序列方式提交待检测的视频。

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acces
sKey Secret")
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   frame1 := map[string]interface{}{
      "offset": "0", "url": "您的视频截帧链接地址1",
   }
   frame2 := map[string]interface{}{
      "offset": "5", "url": "您的视频截帧链接地址2",
   }
   frame3 := map[string]interface{}{
      "offset": "10", "url": "您的视频截帧链接地址3",
   }
   // frames：截帧信息。
   task := map[string]interface{}{"dataId": "检测数据ID", "frames": [...]map[string]interfac
e{}{frame1, frame2, frame3}}
   // scenes：检测场景，支持指定多个场景。
   content, _ := json.Marshal(
      map[string]interface{}{
         "tasks": [...]map[string]interface{}{task}, "scenes": [...]string{"porn", "terrori
sm"},
         "bizType": "业务场景",
      },
   )
   request := green.CreateVideoSyncScanRequest()
   request.SetContent(content)
   response, _err := client.VideoSyncScan(request)
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

视频检测结果反馈

如果您认为视频检测结果与您的预期不符，可以通过视频检测结果反馈接口，对检测结果进行纠正（系统会根据您反馈的结果，将视频截帧添加到相似图片的黑名单库或者白名单库）。当您再次提交相似的内容进行检测时，以您反馈的label返回结果。

关于接口的说明，请参见检测结果反馈。

接口	描述	支持的Region
VideoFeedbackRequest	提交视频检测结果的反馈，以人工反馈的检测结果纠正算法检测结果。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, _err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acce
ssKey Secret")
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   frame1 := map[string]interface{}{"url": "已检测视频截帧链接地址1", "offset": "offset时间戳1"
}
   frame2 := map[string]interface{}{"url": "已检测视频截帧链接地址2", "offset": "offset时间戳2"
}
   frames := [...]map[string]interface{}{frame1, frame2}
   // scenes：检测场景，支持指定多个场景。
   // suggestion：期望的检测结果；pass：正常；block：违规。
   content, _ := json.Marshal(
      map[string]interface{}{
         "taskId": "视频审核任务ID", "dataId": "检测数据ID", "url": "视频链接地址", "frames": f
rames,
         "suggestion": "block", "scenes": [...]string{"ad", "terrorism"}, "note": "备注信息"
,
      },
   )
   request := green.CreateVideoFeedbackRequest()
   request.SetContent(content)
   response, err := client.VideoFeedback(request)
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

5.3.3. 文本反垃圾检测

本文介绍了如何使用Go SDK文本反垃圾接口，对文本内容进行色情、暴恐、涉政等风险进行识别。

功能描述

文本反垃圾接口目前仅支持同步检测。关于参数的详细说明，请参见文本同步检测。

一次请求可以检测多条文本，也可以检测单条文本。按实际检测的文本条数进行计费，请参见计费方式概述。

前提条件

已安装Go依赖。关于安装Go依赖的具体操作，请参见安装Go依赖。

> **说明** 请一定按照安装Go依赖页面中的版本安装，否则会导致调用失败。

文本内容检测

文本垃圾检测支持自定义关键词，例如添加一些竞品关键词等。如果被检测的文本中包含您添加的关键词，算法会返回您block。

您可以前往内容安全控制台添加关键词，也可以通过API接口添加关键词。

接口	描述	支持的Region
TextScanRequest	提交文本反垃圾检测任务，检测场景参数请传递antispam（scenes=antispam）。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, _err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acce
ssKey Secret")
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   task := map[string]interface{}{"content": "待检测文本内容"}
   // scenes：检测场景，唯一取值：antispam。
   content, _ := json.Marshal(
      map[string]interface{}{
         "scenes": [...]string{"antispam"},
         "tasks":  [...]map[string]interface{}{task},
      },
   )
   textScanRequest := green.CreateTextScanRequest()
   textScanRequest.SetContent(content)
   textScanResponse, err := client.TextScan(textScanRequest)
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   if textScanResponse.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(textScanResponse.GetHttpSt
atus()))
   }
   fmt.Println(textScanResponse.GetHttpContentString())
}
```

文本反垃圾结果反馈

如果您认为文本检测的结果与您的期望不符，您可以通过文本反垃圾结果反馈接口纠正算法的检测结果。

系统会将您反馈的结果添加到对应的检测文本库，在您下次提交相似的内容进行检测时，以您通过反馈接口校正后的结果作为检测结果。

接口	描述	支持的地域
TextFeedbackRequest	提交文本反垃圾检测结果的反馈，以人工反馈的检测结果纠正算法检测结果。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
package main
import (
    "encoding/json"
    "fmt"
    "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
    "strconv"
)
func main() {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    client, _err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acce
ssKey Secret")
    if _err != nil {
        fmt.Println(_err.Error())
        return
    }
    // label：反馈的分类，与具体的scene对应。
    content, _ := json.Marshal(
        map[string]interface{}{
            "taskId": "文本检测任务ID", "content": "文本内容", "label": "spam",
        },
    )
    request := green.CreateTextFeedbackRequest()
    request.SetContent(content)
    response, err := client.TextFeedback(request)
    if err != nil {
        fmt.Println(err.Error())
        return
    }
    if response.GetHttpStatus() != 200 {
        fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
    }
    fmt.Println(response.GetHttpContentString())
}
```

5.3.4. 语音反垃圾检测

本文介绍了如何使用Go SDK语音反垃圾接口，检测实时语音流或语音文件中的垃圾内容。

功能描述

语音流检测和语音文件检测均为异步检测，检测结果需要您以轮询或者回调的方式获取。关于调用请求中的检测场景参数scenes，返回结果中的分类参数label，以及操作建议参数suggestion的说明，请参见语音异步检测。

语音检测按照检测的语音文件、语音流的时间长度进行计费，计费粒度为分钟，每天累计检测总时长进行计量统计，每天检测总时长不足一分钟的按照一分钟进行计费。

前提条件

已安装Go依赖。关于安装Go依赖的具体操作，请参见安装Go依赖。

> **说明** 请一定按照安装Go依赖页面中的版本安装，否则会导致调用失败。

提交语音异步检测任务

接口	描述	支持的地域
VoiceAsyncScanRequest	异步检测语音流或语音文件中是否包含违规内容。语音流格式支持： • RTMP • HTTP • FLV	cn-shanghai：华东2（上海）

示例代码

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acces
sKey Secret")
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   task := map[string]interface{}{"dataId": "检测数据ID", "url": "待检测音频链接地址"}
   // scenes：检测场景，唯一取值：antispam。
   content, _ := json.Marshal(
      map[string]interface{}{
         "tasks": [...]map[string]interface{}{task}, "scenes": [...]string{"antispam"},
         "bizType": "业务场景",
      },
   )
   request := green.CreateVoiceAsyncScanRequest()
   request.SetContent(content)
   response, _err := client.VoiceAsyncScan(request)
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

查询语音异步检测结果

接口	描述	支持的地域
VoiceAsyncScanResultsRequest	查询异步语音检测结果。	cn-shanghai：华东2（上海）

示例代码

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acces
sKey Secret")
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   content, _ := json.Marshal([...]string{"语音异步检测任务ID"})
   request := green.CreateVoiceAsyncScanResultsRequest()
   request.SetContent(content)
   response, _err := client.VoiceAsyncScanResults(request)
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

短语音同步检测

接口	描述	支持的地域
VoicesyncscanRequest	查询短语音检测结果。	cn-shanghai：华东2（上海）

示例代码

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acces
sKey Secret")
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   task := map[string]interface{}{"dataId": "检测数据ID", "url": "待检测音频链接地址"}
   // scenes：检测场景，唯一取值：antispam。
   content, _ := json.Marshal(
      map[string]interface{}{
         "tasks": [...]map[string]interface{}{task}, "scenes": [...]string{"antispam"},
         "bizType": "业务场景",
      },
   )
   request := green.CreateVoiceSyncScanRequest()
   request.SetContent(content)
   response, _err := client.VoiceSyncScan(request)
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

5.3.5. 文件审核

本文介绍如何使用Go SDK文件审核接口，检测文件中的文字和图片信息。

功能描述

文件审核目前只支持异步检测（异步检测不会实时返回检测结果）任务。关于参数的详细说明，请参见提交文件检测任务。

前提条件

已安装Go依赖。关于安装Go依赖的具体操作，请参见安装Go依赖。

> **说明** 请一定按照安装Go依赖页面中的版本安装，否则会导致调用失败。

提交文件异步检测任务

接口	描述	支持的地域
FileAsyncScanRequest	提交文件异步检测任务，对文件中的文字和图片进行检测。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京）

示例代码

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acces
sKey Secret")
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   task := map[string]interface{}{
      "dataId": "检测数据ID", "url": "待检测文件链接地址",
   }
   // scenes：检测场景，支持指定多个场景。
   content, _ := json.Marshal(
      map[string]interface{}{
         "textScenes": [...]string{"antispam"}, "imageScenes": [...]string{"porn", "terrori
sm"},
         "tasks": [...]map[string]interface{}{task}, "bizType": "业务场景",
      },
   )
   request := green.CreateFileAsyncScanRequest()
   request.SetContent(content)
   response, _err := client.FileAsyncScan(request)
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

获取文件异步检测任务结果

接口	描述	支持的Region
FileAsyncScanResultsRequest	获取文件异步检测结果。	• **cn-shanghai**：华东2（上海） • **cn-beijing**：华北2（北京）

示例代码

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acces
sKey Secret")
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   content, _ := json.Marshal([...]string{"文件异步检测任务ID"})
   request := green.CreateFileAsyncScanResultsRequest()
   request.SetContent(content)
   response, _err := client.FileAsyncScanResults(request)
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

5.3.6. 网页审核

本文介绍如何使用Go SDK网页审核接口，检测网页中的文字和图片信息。

功能描述

图片审核支持同步检测和异步检测两种方式。

- 同步检测实时返回检测结果。关于参数的详细信息，请参见网页同步检测。
- 异步检测需要您轮询结果或者通过callback回调通知获取检测结果。关于参数的详细信息，请参见网页异步检测。

前提条件

已安装Go依赖。关于安装Go依赖的具体操作，请参见安装Go依赖。

> **说明** 请一定按照安装Go依赖页面中的版本安装，否则会导致调用失败。

提交网页同步检测任务

接口	描述	支持的地域
WebpageSyncScanRequest	提交网页同步检测任务，对网页中的文字和图片进行检测。	• **cn-shanghai**：华东2（上海） • **cn-beijing**：华北2（北京）

示例代码

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acces
sKey Secret")
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   task := map[string]interface{}{"dataId": "检测数据ID", "url": "待检测网页地址链接地址"}
   content, _ := json.Marshal(
      map[string]interface{}{
         "textScenes": [...]string{"antispam"}, "imageScenes": [...]string{"porn", "terrori
sm"},
         "tasks": [...]map[string]interface{}{task}, "bizType": "业务场景",
      },
   )
   request := green.CreateWebpageSyncScanRequest()
   request.SetContent(content)
   response, _err := client.WebpageSyncScan(request)
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

提交网页异步检测任务

接口	描述	支持的Region
WebpageAsyncScanRequest	提交网页异步检测任务，对网页中的文字和图片进行检测。	• **cn-shanghai**：华东2（上海） • **cn-beijing**：华北2（北京）

示例代码

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acces
sKey Secret")
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   task := map[string]interface{}{"dataId": "检测数据ID", "url": "待检测网页地址链接地址"}
   content, _ := json.Marshal(
      map[string]interface{}{
         "textScenes": [...]string{"antispam"}, "imageScenes": [...]string{"porn", "terrori
sm"},
         "tasks": [...]map[string]interface{}{task}, "bizType": "业务场景",
      },
   )
   request := green.CreateWebpageAsyncScanRequest()
   request.SetContent(content)
   response, _err := client.WebpageAsyncScan(request)
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

获取网页异步检测任务结果

接口	描述	支持的Region
WebpageAsyncScanResultsRequest	获取网页异步检测结果。	• **cn-shanghai**：华东2（上海） • **cn-beijing**：华北2（北京）

示例代码

```
package main
import (
    "encoding/json"
    "fmt"
    "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
    "strconv"
)
func main() {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acces
sKey Secret")
    if err != nil {
        fmt.Println(err.Error())
        return
    }
    content, _ := json.Marshal([...]string{"网页异步检测任务ID"})
    request := green.CreateWebpageAsyncScanResultsRequest()
    request.SetContent(content)
    response, _err := client.WebpageAsyncScanResults(request)
    if _err != nil {
        fmt.Println(_err.Error())
        return
    }
    if response.GetHttpStatus() != 200 {
        fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
    }
    fmt.Println(response.GetHttpContentString())
}
```

5.4. 人工审核

5.4.1. 图片人工审核

本文介绍如何使用Go SDK图片人工审核接口，进行人工审核。

功能描述

如果您认为图片检测结果与您的预期不符，可以进行人工审核。关于参数的详细信息，请参见图片人工审核。

关于该API的接入地址，请参见接入地址（Endpoint）。

前提条件

已安装Go依赖。关于安装Go依赖的具体操作，请参见安装Go依赖。

> **说明** 请一定按照安装Go依赖页面中的版本安装，否则会导致调用失败。

提交图片人工审核任务

示例代码

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, _err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acce
ssKey Secret")
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   task := map[string]interface{}{"dataId": "检测数据ID", "url": "待检测图片链接地址"}
   // callback、seed用于回调通知，可选参数。
   content, _ := json.Marshal(
      map[string]interface{}{
         "tasks":    [...]map[string]interface{}{task},
         "callback": "回调地址", "seed": "随机字符串",
      },
   )
   request := green.CreateImageAsyncManualScanRequest()
   request.SetContent(content)
   response, err := client.ImageAsyncManualScan(request)
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

查询图片人工审核任务结果

示例代码

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, _err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acce
ssKey Secret")
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   content, _ := json.Marshal([...]string{"图片人工审核任务ID"})
   request := green.CreateImageAsyncManualScanResultsRequest()
   request.SetContent(content)
   response, err := client.ImageAsyncManualScanResults(request)
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

5.4.2. 视频人工审核

本文介绍如何使用Go SDK视频人工审核接口，进行人工审核。

功能描述

如果您认为视频检测结果与您的预期不符，可以进行人工审核。关于参数的详细信息，请参见视频人工审核API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装Go依赖。关于安装Go依赖的具体操作，请参见安装Go依赖。

> **说明** 请一定按照安装Go依赖页面中的版本安装，否则会导致调用失败。

提交视频人工审核任务

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   client, _err := green.NewClientWithAccessKey("cn-shanghai", "LTAIRwTBJBOT****", "Wo49fNo
S8kbdf38txTFeSuKm8qcbZj")
   // 请替换成您的AccessKey ID、AccessKey Secret。
   // client, _err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的A
ccessKey Secret")
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   task := map[string]interface{}{"dataId": "检测数据ID", "url": "待检测视频链接地址"}
   // callback、seed用于回调通知，可选参数。
   content, _ := json.Marshal(
      map[string]interface{}{
         "tasks":    [...]map[string]interface{}{task},
         "callback": "回调地址", "seed": "随机字符串",
      },
   )
   request := green.CreateVideoAsyncManualScanRequest()
   request.SetContent(content)
   response, err := client.VideoAsyncManualScan(request)
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

查询视频人工审核任务结果

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, _err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acce
ssKey Secret")
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   content, _ := json.Marshal([...]string{"视频人工审核任务ID"})
   request := green.CreateVideoAsyncManualScanResultsRequest()
   request.SetContent(content)
   response, err := client.VideoAsyncManualScanResults(request)
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

5.4.3. 文本人工审核

本文介绍如何使用Go SDK文本人工审核接口，进行人工审核。

功能描述

如果您认为文本检测结果与您的预期不符，可以进行人工审核。关于参数的详细信息，请参见文本人工审核API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装Go依赖。关于安装Go依赖的具体操作，请参见安装Go依赖。

> **说明** 请一定按照安装Go依赖页面中的版本安装，否则会导致调用失败。

提交文本人工审核任务

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, _err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acce
ssKey Secret")
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   task := map[string]interface{}{"dataId": "检测数据ID", "content": "待检测文本"}
   // callback、seed用于回调通知，可选参数。
   content, _ := json.Marshal(
      map[string]interface{}{
         "tasks":    [...]map[string]interface{}{task},
         "callback": "回调地址", "seed": "随机字符串",
      },
   )
   request := green.CreateTextAsyncManualScanRequest()
   request.SetContent(content)
   response, err := client.TextAsyncManualScan(request)
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

查询文本人工审核任务结果

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, _err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acce
ssKey Secret")
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   content, _ := json.Marshal([...]string{"文本人工审核任务ID"})
   request := green.CreateTextAsyncManualScanResultsRequest()
   request.SetContent(content)
   response, err := client.TextAsyncManualScanResults(request)
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

5.4.4. 语音人工审核

本文介绍如何使用Go SDK语音人工审核接口，进行人工审核。

功能描述

如果您认为语音检测结果与您的预期不符，可以进行人工审核。关于参数的详细信息，请参见语音人工审核API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装Go依赖。关于安装Go依赖的具体操作，请参见安装Go依赖。

> **说明** 请一定按照安装Go依赖页面中的版本安装，否则会导致调用失败。

提交语音人工审核任务

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, _err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acce
ssKey Secret")
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   task := map[string]interface{}{"dataId": "检测数据ID", "url": "待检测音频链接地址"}
   // callback、seed用于回调通知，可选参数。
   content, _ := json.Marshal(
      map[string]interface{}{
         "tasks":    [...]map[string]interface{}{task},
         "callback": "回调地址", "seed": "随机字符串",
      },
   )
   request := green.CreateVoiceAsyncManualScanRequest()
   request.SetContent(content)
   response, err := client.VoiceAsyncManualScan(request)
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

查询语音人工审核任务结果

```
package main
import (
    "encoding/json"
    "fmt"
    "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
    "strconv"
)
func main() {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    client, _err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acce
ssKey Secret")
    if _err != nil {
        fmt.Println(_err.Error())
        return
    }
    content, _ := json.Marshal([...]string{"音频人工审核任务ID"})
    request := green.CreateVoiceAsyncManualScanResultsRequest()
    request.SetContent(content)
    response, err := client.VoiceAsyncManualScanResults(request)
    if err != nil {
        fmt.Println(err.Error())
        return
    }
    if response.GetHttpStatus() != 200 {
        fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
    }
    fmt.Println(response.GetHttpContentString())
}
```

5.5. 图片OCR识别

本文介绍了如何使用Go SDK图片OCR接口，识别图片中的文字或卡证信息。

功能描述

通用OCR除了能够识别普通图片中的文字，还能识别结构化的卡证上的文字。关于参数的详细说明，请参见图片OCR检测API文档。

前提条件

已安装Go依赖。关于安装Go依赖的具体操作，请参见安装Go依赖。

> **说明** 请一定按照安装Go依赖页面中的版本安装，否则会导致调用失败。

提交图片同步检测任务

接口	描述	支持的Region

接口	描述	支持的Region
ImageSyncScanRequest	提交图片OCR同步识别任务，对图片中的文字进行识别（scene=ocr）。	• *cn-shanghai* • *cn-beijing* • *cn-shenzhen* • *ap-southeast-1*

示例代码

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acces
sKey Secret")
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   task1 := map[string]interface{}{"dataId": "检测数据ID", "url": "待检测图片链接地址"}
   // 示例：身份证正面识别。
   cardExtras := map[string]interface{}{"card": "id-card-front"}
   // 示例：身份证反面识别。
   // cardExtras := map[string]interface{}{"card": "id-card-back"}
   // scenes：检测场景。
   content, _ := json.Marshal(
      map[string]interface{}{
         "tasks": task1, "scenes": [...]string{"ocr"}, "bizType": "业务场景", "extras": card
Extras,
      },
   )
   request := green.CreateImageSyncScanRequest()
   request.SetContent(content)
   response, _err := client.ImageSyncScan(request)
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

5.6. 人脸识别

5.6.1. 人脸比对

本文介绍了如何使用Go SDK人脸比对接口，对指定的人脸图片进行属性检测。人脸属性检测仅支持以图片作为检测媒介。

功能描述

人脸比对支持同步检测和异步检测两种方式。

- 同步检测实时返回检测结果。关于参数的详细信息，请参见同步检测。
- 异步检测需要您轮询结果或者通过callback回调通知获取检测结果。关于参数的详细信息，请参见异步检测。

前提条件

已安装Go依赖。关于安装Go依赖的具体操作，请参见安装Go依赖。

> **说明** 请一定按照安装Go依赖页面中的版本安装，否则会导致调用失败。

提交人脸比对任务

```
ackage main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acces
sKey Secret")
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   // 设置待比对人脸图片链接地址2。
   extras := map[string]interface{}{"faceUrl": "待比对人脸图片链接地址2"}
   // 设置待比对人脸图片链接地址1。
   task1 := map[string]interface{}{"dataId": "dataIdxxx", "url": "待比对人脸图片链接地址1", "e
xtras": extras}
  // scenes：检测场景，指定图片检测的应用场景，取值：sface-1。
   content, _ := json.Marshal(
      map[string]interface{}{
         "tasks": task1, "scenes": [...]string{"sface-1"}, "bizType": "业务场景",
      },
   )
   request := green.CreateImageSyncScanRequest()
   request.SetContent(content)
   response, _err := client.ImageSyncScan(request)
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

5.6.2. 人脸属性检测

本文介绍了如何使用Go SDK人脸属性检测接口，对指定的人脸图片（公网图片）进行属性检测。人脸属性检测仅支持以图片作为检测媒介。

功能描述

人脸属性检测能够识别图片中的人脸属性信息，包括人脸模糊度、人脸角度、人脸位置、微笑程度、是否戴眼镜、是否戴口罩、是否戴帽子、是否有胡子、是否有刘海、头发类型等。关于参数的详细说明，请参见人脸属性检测API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装Go依赖。关于安装Go依赖的具体操作，请参见安装Go依赖。

> **说明** 请一定按照安装Go依赖页面中的版本安装，否则会导致调用失败。

提交人脸属性检测任务

```
package main
import (
    "encoding/json"
    "fmt"
    "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
    "strconv"
)
func main() {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acces
sKey Secret")
    if err != nil {
        fmt.Println(err.Error())
        return
    }
    content, _ := json.Marshal(
        map[string]interface{}{
            "url": "待检测人脸图片链接地址",
        },
    )
    request := green.CreateDetectFaceRequest()
    request.SetContent(content)
    response, _err := client.DetectFace(request)
    if _err != nil {
        fmt.Println(_err.Error())
        return
    }
    if response.GetHttpStatus() != 200 {
        fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
    }
    fmt.Println(response.GetHttpContentString())
}
```

5.6.3. 自定义人脸检索

5.6.3.1. 自定义人脸检索

本文介绍了如何使用Go SDK自定义人脸检索接口，对指定的人脸图片进行自定义人脸检索。

功能描述

自定义人脸检索根据您输入的待识别人脸图片（face），在指定人脸库（group）中查找并返回最相似的Top 5个体（person），返回的Top 5个体按照相似度从大到小排序。

检索人脸时，必须指定要检索的分组信息（最多指定一个分组）。关于参数的详细说明，请参见自定义人脸检索API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装Go依赖。关于安装Go依赖的具体操作，请参见安装Go依赖。

> **说明** 请一定按照安装Go依赖页面中的版本安装，否则会导致调用失败。

提交人脸图片检索任务

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acces
sKey Secret")
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   extras := map[string]interface{}{"groupId": "个体组ID"}
   task1 := map[string]interface{}{"dataId": "检测数据ID", "url": "待检测人脸图片链接地址", "ex
tras": extras}
   // scenes：检测场景，取值：sface-n，表示自定义人脸检索。
   content, _ := json.Marshal(
      map[string]interface{}{
         "tasks": task1, "scenes": [...]string{"sface-n"}, "bizType": "业务场景",
      },
   )
   request := green.CreateImageSyncScanRequest()
   request.SetContent(content)
   response, _err := client.ImageSyncScan(request)
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

5.6.3.2. 新建个体

本文介绍了如何使用Go SDK新建个体信息。

功能描述

新建个体时，必须指定个体要加入的分组。关于参数的详细说明，请参见新建个体API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装Go依赖。关于安装Go依赖的具体操作，请参见安装Go依赖。

> **说明** 请一定按照安装Go依赖页面中的版本安装，否则会导致调用失败。

新建个体任务

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acces
sKey Secret")
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   /**
    * personId: 用户自定义个体ID，必填。
    * groupIds: 用户自定义组ID列表，必填。
    * name: 用户名称，非必填。
    * note: 备注信息，非必填。
    */
   content, _ := json.Marshal(
      map[string]interface{}{
         "personId": "个体ID", "groupIds": [...]string{"组ID_1","组ID_2"}, "name": "名称",
         "note": "备注信息",
      },
   )
   request := green.CreateAddPersonRequest()
   request.SetContent(content)
   response, _err := client.AddPerson(request)
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

5.6.3.3. 设置个体

本文介绍了如何使用Go SDK设置个体信息。

功能描述

设置个体信息用于给个体添加备注信息，属于可选步骤。如果您设置了个体信息，则在自定义检索的返回结果中会包含该信息。关于参数的详细说明，请参见设置个体API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装Go依赖。关于安装Go依赖的具体操作，请参见安装Go依赖。

> **说明** 请一定按照安装Go依赖页面中的版本安装，否则会导致调用失败。

提交设置个体任务

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acces
sKey Secret")
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   /**
    * personId: 用户自定义个体ID，必填。
    * name: 用户名称，非必填。
    * note: 备注信息，非必填。
    */
   content, _ := json.Marshal(
      map[string]interface{}{
         "personId": "个体ID", "name": "名称", "note": "备注信息",
      },
   )
   request := green.CreateSetPersonRequest()
   request.SetContent(content)
   response, _err := client.SetPerson(request)
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

5.6.3.4. 查询个体信息

本文介绍了如何使用Go SDK查询个体信息。

功能描述

关于参数的详细说明，请参见查询个体信息API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装Go依赖。关于安装Go依赖的具体操作，请参见安装Go依赖。

> **说明** 请一定按照安装Go依赖页面中的版本安装，否则会导致调用失败。

查询个体信息任务

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acces
sKey Secret")
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   /**
    * personId：用户自定义个体ID，必填。
    */
   content, _ := json.Marshal(
      map[string]interface{}{
         "personId": "个体ID",
      },
   )
   request := green.CreateGetPersonRequest()
   request.SetContent(content)
   response, _err := client.GetPerson(request)
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

5.6.3.5. 查询人脸信息

本文介绍了如何使用Go SDK查询人脸信息。

功能描述

根据个体ID和人脸ID查询当前个体的人脸ID以及人脸图片路径等信息。关于查询人脸的具体参数说明，请参见查询人脸图片API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装Go依赖。关于安装Go依赖的具体操作，请参见安装Go依赖。

> **说明** 请一定按照安装Go依赖页面中的版本安装，否则会导致调用失败。

查询人脸信息任务

```
package main
import (
    "encoding/json"
    "fmt"
    "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
    "strconv"
)
func main() {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acces
sKey Secret")
    if err != nil {
        fmt.Println(err.Error())
        return
    }
    /**
     * personId: 用户自定义个体ID，必填。
     * faceIds: 人脸ID集合，必填。
     */
    content, _ := json.Marshal(
        map[string]interface{}{
            "personId": "个体ID",
            "faceIds":  [...]string{"人脸ID_1", "人脸ID_2"},
        },
    )
    request := green.CreateGetFacesRequest()
    request.SetContent(content)
    response, _err := client.GetFaces(request)
    if _err != nil {
        fmt.Println(_err.Error())
        return
    }
    if response.GetHttpStatus() != 200 {
        fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
    }
    fmt.Println(response.GetHttpContentString())
}
```

5.6.3.6. 删除个体

本文介绍了如何使用.NET SDK删除个体信息。

功能描述

删除个体时，个体对应的图片以及信息均会被删除。关于参数的详细说明，请参见删除个体API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装Go依赖。关于安装Go依赖的具体操作，请参见安装Go依赖。

> **说明** 请一定按照安装Go依赖页面中的版本安装，否则会导致调用失败。

删除个体任务

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   // client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Ac
cessKey Secret")
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   /**
    * personId：用户自定义个体ID，必填。
    */
   content, _ := json.Marshal(
      map[string]interface{}{
         "personId": "个体ID",
      },
   )
   request := green.CreateDeletePersonRequest()
   request.SetContent(content)
   response, _err := client.DeletePerson(request)
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

5.6.3.7. 新增人脸

本文介绍了如何使用Go SDK新增个体的人脸图片。

功能描述

为个体新增一组人脸图片，一个个体最多允许包含20张人脸图片。关于参数的说明，请参见新增人脸API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装Go依赖。关于安装Go依赖的具体操作，请参见安装Go依赖。

> **说明** 请一定按照安装Go依赖页面中的版本安装，否则会导致调用失败。

新增人脸图片任务

```
package main
import (
    "encoding/json"
    "fmt"
    "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
    "strconv"
)
func main() {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acces
sKey Secret")
    if err != nil {
        fmt.Println(err.Error())
        return
    }
    /**
     * personId: 用户自定义个体ID，必填。
     * urls: 人脸图片URL。
     */
    content, _ := json.Marshal(
        map[string]interface{}{
            "personId": "个体ID", "urls": [...]string{"人脸图片链接地址"},
        },
    )
    request := green.CreateAddFacesRequest()
    request.SetContent(content)
    response, _err := client.AddFaces(request)
    if _err != nil {
        fmt.Println(_err.Error())
        return
    }
    if response.GetHttpStatus() != 200 {
        fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
    }
    fmt.Println(response.GetHttpContentString())
}
```

5.6.3.8. 删除人脸

本文介绍了如何使用Go SDK，删除无用的个体人脸图片。

功能描述

删除人脸图片时，必须指定要删除的图片ID以及对应的个体ID信息。关于参数的详细说明，请参见删除人脸API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装Go依赖。关于安装Go依赖的具体操作，请参见安装Go依赖。

> **说明** 请一定按照安装Go依赖页面中的版本安装，否则会导致调用失败。

提交删除人脸任务

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acces
sKey Secret")
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   /**
    * personId: 用户自定义个体ID，必填。
    * faceIds: 已添加的人脸ID。
    */
   content, _ := json.Marshal(
      map[string]interface{}{
         "personId": "个体ID",
         "faceIds":  [...]string{"人脸ID_1"},
      },
   )
   request := green.CreateDeleteFacesRequest()
   request.SetContent(content)
   response, _err := client.DeleteFaces(request)
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

5.6.3.9. 查询组列表

本文介绍了如何使用Go SDK查询所有人脸分组ID。

功能描述

查询所有的个体组ID。关于参数的详细说明，请参见查询组列表API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装Go依赖。关于安装Go依赖的具体操作，请参见安装Go依赖。

> **说明** 请一定按照安装Go依赖页面中的版本安装，否则会导致调用失败。

查询所有分组ID任务

```
package main
import (
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acces
sKey Secret")
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   request := green.CreateGetGroupsRequest()
   response, _err := client.GetGroups(request)
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

5.6.3.10. 查询个体列表

本文介绍了如何使用Go SDK查询个体列表信息。

功能描述

每个个体必须属于一个分组，在调用该接口时指定某个分组，则返回该分组下的所有个体ID列表。关于参数的详细说明，请参见查询个体列表API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装Go依赖。关于安装Go依赖的具体操作，请参见安装Go依赖。

> **说明** 请一定按照安装Go依赖页面中的版本安装，否则会导致调用失败。

查询个体列表任务

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acces
sKey Secret")
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   /**
    * groupId: 用户自定义组ID，必填。
    */
   content, _ := json.Marshal(
      map[string]interface{}{
         "groupId": "组ID",
      },
   )
   request := green.CreateGetPersonsRequest()
   request.SetContent(content)
   response, _err := client.GetPersons(request)
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

5.6.3.11. 添加个体到分组

本文介绍了如何使用Go SDK添加个体到指定分组。

功能描述

将一个个体添加到一个或者多个个体组。关于参数的详细说明，请参见添加个体到分组API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装Go依赖。关于安装Go依赖的具体操作，请参见安装Go依赖。

> **说明** 请一定按照安装Go依赖页面中的版本安装，否则会导致调用失败。

添加个体到分组任务

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acces
sKey Secret")
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   /**
    * personId: 用户自定义个体ID，必填。
    * groupIds: 用户自定义组ID列表，必填。
    */
   content, _ := json.Marshal(
      map[string]interface{}{
         "personId": "个体ID", "groupIds": [...]string{"组ID_1"},
      },
   )
   request := green.CreateAddGroupsRequest()
   request.SetContent(content)
   response, _err := client.AddGroups(request)
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

5.6.3.12. 删除分组中个体

本文介绍了如何使用Go SDK从指定分组中删除个体信息。

功能描述

删除分组中个体不会删除个体对应的信息以及图片，只是解除个体与该分组的绑定关系。关于参数的详细说明，请参见删除分组中的个体API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装Go依赖。关于安装Go依赖的具体操作，请参见安装Go依赖。

> **说明** 请一定按照安装Go依赖页面中的版本安装，否则会导致调用失败。

从指定分组删除个体任务

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acces
sKey Secret")
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   /**
    * personId: 用户自定义个体ID，必填。
    * groupIds: 用户自定义组ID列表，必填。
    */
   content, _ := json.Marshal(
      map[string]interface{}{
         "personId": "个体ID", "groupIds": [...]string{"组ID_1"},
      },
   )
   request := green.CreateDeleteGroupsRequest()
   request.SetContent(content)
   response, _err := client.DeleteGroups(request)
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

5.7. 相似图检索

5.7.1. 创建相似图样本库

本文介绍了如何使用Go SDK，创建相似图样本库。

功能描述

通过该接口创建相似图样本图库。关于参数的详细说明，请参见创建图库API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装Go依赖。关于安装Go依赖的具体操作，请参见安装Go依赖。

> **说明** 请一定按照安装Go依赖页面中的版本安装，否则会导致调用失败。

创建相似图样本任务

```
package main
import (
    "encoding/json"
    "fmt"
    "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
    "strconv"
)
func main() {
    // 请替换成您的AccessKey ID、AccessKey Secret。
    client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acces
sKey Secret")
    if err != nil {
        fmt.Println(err.Error())
        return
    }
    content, _ := json.Marshal(
        map[string]interface{}{
            "name": "相似图样本图库名称",
        },
    )
    request := green.CreateAddSimilarityLibraryRequest()
    request.SetContent(content)
    response, _err := client.AddSimilarityLibrary(request)
    if _err != nil {
        fmt.Println(_err.Error())
        return
    }
    if response.GetHttpStatus() != 200 {
        fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
    }
    fmt.Println(response.GetHttpContentString())
}
```

5.7.2. 相似图检索

本文介绍了如何使用Go SDK，对指定的图片进行相似图检索。

功能描述

相似图检索帮助您从图库中检索出与给定的图片相似的若干张图片。关于参数的详细说明，请参见相似图检索API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装Go依赖。关于安装Go依赖的具体操作，请参见安装Go依赖。

> **说明** 请一定按照安装Go依赖页面中的版本安装，否则会导致调用失败。

提交相似图检索任务

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acces
sKey Secret")
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   task1 := map[string]interface{}{
      "dataId":               "检测数据ID",
      "url":                  "待检测图片链接",
      "similarityLibraries": [...]string{"相似图样本图库名称"},
   }
   /**
    * 设置要检测的场景，计费是按照该处传递的场景进行。
    * similarity: similarity表示进行相似图检索。
    */
   content, _ := json.Marshal(
      map[string]interface{}{
         "tasks": task1, "scenes": [...]string{"similarity"},
      },
   )
   request := green.CreateImageSyncScanRequest()
   request.SetContent(content)
   response, _err := client.ImageSyncScan(request)
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

5.7.3. 增加相似图样本

本文介绍了如何使用Go SDK，将指定的相似图图片添加到对应图库。

功能描述

添加相似图样本时，您可以为图片设置标签。如果样本图片在检索时被命中，其标签信息也会被返回。关于参数的详细说明，请参见增加样本图片API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装Go依赖。关于安装Go依赖的具体操作，请参见安装Go依赖。

> **说明** 请一定按照安装Go依赖页面中的版本安装，否则会导致调用失败。

增加相似图样本任务

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acces
sKey Secret")
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   task := map[string]interface{}{
      "dataId": "检测数据ID",
      "url":    "样本图片链接",
      "tag":    [...]string{"自定义标签1", "自定义标签2", "自定义标签3"},
   }
   content, _ := json.Marshal(
      map[string]interface{}{
         "library": "相似图样本图库名称", "tasks": [...]map[string]interface{}{task},
      },
   )
   request := green.CreateAddSimilarityImageRequest()
   request.SetContent(content)
   response, _err := client.AddSimilarityImage(request)
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

5.7.4. 查询相似图库列表

本文介绍了如何使用Go SDK，分页查询相似图库及其元素信息。

功能描述

调用该接口分页查询所有相似图库及其元素信息。关于参数的详细说明，请参见查询相似图库列表API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装Go依赖。关于安装Go依赖的具体操作，请参见安装Go依赖。

> **说明** 请一定按照安装Go依赖页面中的版本安装，否则会导致调用失败。

查询相似图库列表任务

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您自的AccessKey ID、AccessKey Secret。
   client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acces
sKey Secret")
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   // pageSize：每页大小，取值范围：(0,50]；currentPage：当前页码，取值范围：(0,50]。
   content, _ := json.Marshal(
      map[string]interface{}{
         "pageSize": "5", "currentPage": 1,
      },
   )
   request := green.CreateListSimilarityLibrariesRequest()
   request.SetContent(content)
   response, _err := client.ListSimilarityLibraries(request)
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

5.7.5. 查询指定相似图库信息

本文介绍了如何使用Go SDK，查询指定相似图库信息。

功能描述

调用该接口查询指定相似图库信息。关于参数的详细说明，请参见查询指定相似图库API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装Go依赖。关于安装Go依赖的具体操作，请参见安装Go依赖。

> **说明** 请一定按照安装Go依赖页面中的版本安装，否则会导致调用失败。

查询指定相似图库任务

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acces
sKey Secret")
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   content, _ := json.Marshal(
      map[string]interface{}{
         "name": "相似图样本图库名称",
      },
   )
   request := green.CreateGetSimilarityLibraryRequest()
   request.SetContent(content)
   response, _err := client.GetSimilarityLibrary(request)
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

5.7.6. 查询相似图样本图片列表

本文介绍了如何使用Go SDK，分页查询指定相似图库下的所有样本图片信息。

功能描述

调用该接口分页查询指定相似图库下的所有样本图片信息。关于参数的详细说明，请参见查询样本图片列表API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装Go依赖。关于安装Go依赖的具体操作，请参见安装Go依赖。

> **说明** 请一定按照安装Go依赖页面中的版本安装，否则会导致调用失败。

查询相似图样本图片列表任务

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acces
sKey Secret")
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   //pageSize：每页大小，取值范围：(0,50]；currentPage：当前页码，取值范围：(0,50]
   content, _ := json.Marshal(
      map[string]interface{}{
         "library": "相似图样本图库名称", "pageSize": "5", "currentPage": 1,
      },
   )
   request := green.CreateListSimilarityImagesRequest()
   request.SetContent(content)
   response, _err := client.ListSimilarityImages(request)
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

5.7.7. 查询相似图样本图片详情

本文介绍了如何使用Go SDK，查询相似图样本图片详情。

功能描述

调用该接口查询相似图样本图片详情。关于参数的详细说明，请参见查询样本图库详情API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装Go依赖。关于安装Go依赖的具体操作，请参见安装Go依赖。

> **说明** 请一定按照安装Go依赖页面中的版本安装，否则会导致调用失败。

查询相似图样本图片详情任务

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acces
sKey Secret")
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   content, _ := json.Marshal(
      map[string]interface{}{
         "library": "相似图样本图库名称", "dataId": "样本图片ID",
      },
   )
   request := green.CreateGetSimilarityImageRequest()
   request.SetContent(content)
   response, _err := client.GetSimilarityImage(request)
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

5.7.8. 移除相似图样本

本文介绍了如何使用Go SDK，将相似图样本从图库中移除。

功能描述

移除操作在1分钟之内生效。一次最多允许移除100张样本图片。关于参数的详细说明，请参见删除样本图片API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装Go依赖。关于安装Go依赖的具体操作，请参见安装Go依赖。

> **说明** 请一定按照安装Go依赖页面中的版本安装，否则会导致调用失败。

移除相似图样本任务

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acces
sKey Secret")
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   content, _ := json.Marshal(
      map[string]interface{}{
         "dataIds": [...]string{"样本图片ID 1", "样本图片ID 2"},
      },
   )
   request := green.CreateDeleteSimilarityImageRequest()
   request.SetContent(content)
   response, _err := client.DeleteSimilarityImage(request)
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

5.7.9. 删除相似图库

本文介绍了如何使用Go SDK，删除指定相似图库。

功能描述

只有当相似图库中无图片样本时，才允许被删除。关于参数的详细说明，请参见删除图库API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装Go依赖。关于安装Go依赖的具体操作，请参见安装Go依赖。

> **说明** 请一定按照安装Go依赖页面中的版本安装，否则会导致调用失败。

删除相似图库任务

```
package main
import (
   "encoding/json"
   "fmt"
   "github.com/aliyun/alibaba-cloud-sdk-go/services/green"
   "strconv"
)
func main() {
   // 请替换成您的AccessKey ID、AccessKey Secret。
   client, err := green.NewClientWithAccessKey("cn-shanghai", "您的AccessKey ID", "您的Acces
sKey Secret")
   if err != nil {
      fmt.Println(err.Error())
      return
   }
   content, _ := json.Marshal(
      map[string]interface{}{
         "name": "相似图样本图库名称",
      },
   )
   request := green.CreateDeleteSimilarityLibraryRequest()
   request.SetContent(content)
   response, _err := client.DeleteSimilarityLibrary(request)
   if _err != nil {
      fmt.Println(_err.Error())
      return
   }
   if response.GetHttpStatus() != 200 {
      fmt.Println("response not success. status:" + strconv.Itoa(response.GetHttpStatus()))
   }
   fmt.Println(response.GetHttpContentString())
}
```

6..NET SDK

6.1. 安装

在使用内容检测API .NET SDK前，您需要按照以下步骤安装依赖。

环境准备

- .NET Framework版本为4.5及其以上版本。
- .NET Standard版本为2.0及其以上版本。
- C#版本为4.0及其以上版本。

安装SDK

1. 下载.NET SDK。
2. 安装内容安全.NET SDK。
   - 通过.NET CLI工具安装SDK

     > **说明** 如果您需要安装指定版本，需添加 `--version <具体的版本号>` ，例如 `dotnet add package aliyun-net-sdk-core --version 1.0.0` 。不添加默认安装此软件包的最新版本。

     ```
     dotnet add package aliyun-net-sdk-core
     dotnet add package aliyun-net-sdk-green
     ```

   - 通过 `dotnet` 命令安装
     a. 配置.csproj文件引入依赖包

     ```
     <PackageReference Include="aliyun-net-sdk-core" Version="1.5.10" />
     <PackageReference Include="aliyun-net-sdk-green" Version="3.6.5" />
     ```

     b. 执行以下命令安装SDK

     ```
     dotnet build
     ```

6.2. 初始化

您需要创建一个客户端（Client）实例来发起API请求，并根据需要修改实例的配置参数。

背景信息

新建实例时，需要指定服务地域（Region）和阿里云账号的AccessKey信息（包括AccessKey ID和AccessKey Secret）。

- 关于服务地域的更多信息，请参见接入区域。
- 关于阿里云账号AccessKey的更多信息，请参见获取AccessKey。

创建用于图片、语音、视频、文本、视频识别的客户端实例

支持的地域包括：

- **cn-shanghai**：华东2（上海）

- cn-beijing：华北2（北京）
- cn-shenzhen：华南1（深圳）
- ap-southeast-1：新加坡

您可以参见以下代码创建实例，用于图片、语音、视频、文本检测：

```
// 请替换成您的AccessKey ID、AccessKey Secret。
IClientProfile profile = DefaultProfile.GetProfile("cn-shanghai", "您的AccessKey ID", "您的AccessKey Secret");
DefaultAcsClient client = new DefaultAcsClient(profile);
```

> **说明** 使用其他地域时，请将代码中所有 `cn-shanghai` 标识替换成对应的地域标识，如 `cn-beijing` 。

6.3. 内容审核（机审）

6.3.1. 图片审核

本文介绍了如何使用.NET SDK图片审核接口，检测图片中是否包含风险内容。

功能描述

图片审核支持同步检测和异步检测两种方式。

- 同步检测实时返回检测结果。关于参数的详细信息，请参见同步检测API文档。
- 异步检测需要您轮询结果或者通过callback回调通知获取检测结果。关于参数的详细信息，请参见异步检测API文档。

支持传入检测的图片内容媒介包括互联网图片URL、本地图片文件路径和二进制图片文件流。

前提条件

已安装.NET依赖。关于安装.NET依赖的具体操作，请参见安装.NET依赖。

> **说明** 请一定按照安装.NET依赖页面中的版本安装，否则会导致调用失败。

（推荐）图片同步检测

接口	描述	支持的地域
ImageSyncScanRequest	提交图片同步检测任务，对图片进行多个风险场景的识别，包括色情、暴恐涉政、广告、二维码、不良场景、Logo（商标台标）识别。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            ImageSyncScanRequest request = new ImageSyncScanRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            Dictionary<string, object> task1 = new Dictionary<string, object>();
            task1.Add("dataId", "检测数据ID");
            task1.Add("url", "待检测图片链接地址");
            Dictionary<string, object> httpBody = new Dictionary<string, object>();
            // scenes：检测场景，支持指定多个场景。
            httpBody.Add("scenes", new List<string> { "porn", "terrorism" });
            httpBody.Add("bizType", "业务场景");
            httpBody.Add("tasks", new List<Dictionary<string, object>> { task1 });
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(httpBody)), "utf-8", FormatType.JSON);
            try
            {
                ImageSyncScanResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

提交图片异步检测任务

使用.NET SDK对图片进行风险检测，通过异步请求提交检测任务，结果可以通过提交请求时设置callback回调获取，也可以通过接口轮询获取。

异步检测支持的输入内容与同步检测一致（本地文件、图片URL、图片二进制流），以下仅以图片URL作为输入示例。

接口	描述	支持的地域
ImageAsyncScanRequest	提交图片异步检测任务，对图片进行多个风险场景的识别，包括色情、暴恐涉政、广告、二维码、不良场景、Logo（商标台标）识别。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            ImageAsyncScanRequest request = new ImageAsyncScanRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            Dictionary<string, object> task1 = new Dictionary<string, object>();
            task1.Add("dataId", "检测数据ID");
            task1.Add("url", "待检测图片链接地址");
            Dictionary<string, object> httpBody = new Dictionary<string, object>();
            // scenes：检测场景，支持指定多个场景。
            httpBody.Add("scenes", new List<string> { "porn", "terrorism" });
            httpBody.Add("bizType", "业务场景");
            httpBody.Add("tasks", new List<Dictionary<string, object>> { task1 });
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(httpBody)), "utf-8", FormatType.JSON);
            try
            {
                ImageAsyncScanResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

查询异步检测结果

接口	描述	支持的地域
ImageAsyncScanResultsRequest	查询图片异步检测任务的结果。支持同时查询多个检测任务的返回结果。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                 "cn-shanghai",
                 "您的AccessKey ID",
                 "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            ImageAsyncScanResultsRequest request = new ImageAsyncScanResultsRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            List<string> taskIdList = new List<string> { "图片异步检测任务ID"};
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(taskIdList)), "utf-8", FormatType.JSON);
            try
            {
                ImageAsyncScanResultsResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

图片检测结果反馈

如果您认为图片检测结果与您的预期不符，可以通过图片检测结果反馈接口，对检测结果进行纠正（系统会根据您反馈的结果，将图片添加到相似图片的黑名单库或者白名单库）。当您再次提交相似的内容进行检测时，以您反馈的label返回结果。

关于接口的说明，请参见检测结果反馈。

接口	描述	支持的Region
ImageScanFeedbackRequest	提交图片检测结果的反馈，以人工反馈的检测结果纠正算法检测结果。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            ImageScanFeedbackRequest request = new ImageScanFeedbackRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            // scenes：检测场景，支持指定多个场景。
            // suggestion：期望的检测结果，pass：正常；block：违规。
            Dictionary<string, object> httpBody = new Dictionary<string, object>();
            httpBody.Add("scenes", new List<string> { "porn", "terrorism" });
            httpBody.Add("suggestion", "block");
            httpBody.Add("url", "图片链接地址");
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(httpBody)), "utf-8", FormatType.JSON);
            try
            {
                ImageScanFeedbackResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

6.3.2. 视频审核

本文介绍了如何使用.NET SDK视频审核接口，检测视频中是否包含风险内容。

功能描述

视频审核接口支持同步检测和异步检测两种方式。

- 同步检测只支持传递视频的截帧图片序列。关于参数的详细介绍，请参见同步检测API文档。
- 异步检测（推荐）支持对原始视频或视频的截帧图片序列进行检测。关于参数的详细介绍，请参见异步检测API文档。

前提条件

已安装.NET依赖。关于安装.NET依赖的具体操作，请参见安装.NET依赖。

> **说明** 请一定按照安装.NET依赖页面中的版本安装，否则会导致调用失败。

（推荐）提交视频异步检测任务

接口	描述	支持的地域
VideoAsyncScanRequest	提交视频异步检测任务，对视频进行多个风险场景的识别，包括色情、暴恐涉政、广告 、不良场景、Logo（商标台标）识别。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

- 提交视频URL进行检测

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            VideoAsyncScanRequest request = new VideoAsyncScanRequest();
            request.AcceptFormat = FormatType.JSON;
```

```
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            Dictionary<string, object> task1 = new Dictionary<string, object>();
            task1.Add("dataId", "检测数据ID");
            task1.Add("url", "待检测视频链接地址");
            // scenes：检测场景，支持指定多个场景。
            // callback、seed用于回调通知，可选参数。
            Dictionary<string, object> httpBody = new Dictionary<string, object>();
            httpBody.Add("scenes", new List<string> { "porn", "terrorism" });
            httpBody.Add("bizType", "业务场景");
            httpBody.Add("callback", "回调地址");
            httpBody.Add("seed", "随机字符串");
            httpBody.Add("tasks", new List<Dictionary<string, object>> { task1 });
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.Serializ
eObject(httpBody)), "utf-8", FormatType.JSON);
            try
            {
                VideoAsyncScanResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpResp
onse.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRes
ponse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

- 提交视频直播流进行检测

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                 "cn-shanghai",
                 "您的AccessKey ID",
                 "您的AccessKey Secret");
```

```
            DefaultAcsClient client = new DefaultAcsClient(profile);
            VideoAsyncScanRequest request = new VideoAsyncScanRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            Dictionary<string, object> task1 = new Dictionary<string, object>();
            task1.Add("dataId", "检测数据ID");
            // url填写直播流地址。
            task1.Add("url", "待检测视频链接地址");
            // scenes：检测场景，支持指定多个场景。
            // callback、seed用于回调通知，可选参数。
            Dictionary<string, object> httpBody = new Dictionary<string, object>();
            httpBody.Add("scenes", new List<string> { "porn", "terrorism" });
            httpBody.Add("live", "true");
            httpBody.Add("bizType", "业务场景");
            httpBody.Add("callback", "回调地址");
            httpBody.Add("seed", "随机字符串");
            httpBody.Add("tasks", new List<Dictionary<string, object>> { task1 });
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.Serializ
eObject(httpBody)), "utf-8", FormatType.JSON);
            try
            {
                VideoAsyncScanResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpResp
onse.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRes
ponse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

- 提交视频语音进行综合检测

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
```

```
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                 "cn-shanghai",
                 "您的AccessKey ID",
                 "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            VideoAsyncScanRequest request = new VideoAsyncScanRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            Dictionary<string, object> task1 = new Dictionary<string, object>();
            task1.Add("dataId", "检测数据ID");
            // url填写直播流地址。
            task1.Add("url", "待检测视频链接地址");
            // scenes：检测场景，支持指定多个场景。
            // callback、seed用于回调通知，可选参数。
            Dictionary<string, object> httpBody = new Dictionary<string, object>();
            httpBody.Add("scenes", new List<string> { "porn", "terrorism" });
            httpBody.Add("live", "true");
            httpBody.Add("audioScenes", new List<string> { "antispam" });
            httpBody.Add("bizType", "业务场景");
            httpBody.Add("callback", "回调地址");
            httpBody.Add("seed", "随机字符串");
            httpBody.Add("tasks", new List<Dictionary<string, object>> { task1 });
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.Serializ
eObject(httpBody)), "utf-8", FormatType.JSON);
            try
            {
                VideoAsyncScanResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpResp
onse.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRes
ponse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

查询视频异步检测结果

接口	描述	支持的地域

接口	描述	支持的地域
VideoAsyncScanResultsRequest	查询视频异步检测任务的结果。 **说明** 该方法需要轮询结果，建议使用callback的方式获取结果。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            VideoAsyncScanResultsRequest request = new VideoAsyncScanResultsRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            List<string> taskIdList = new List<string> { "视频异步检测任务ID" };
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(taskIdList)), "utf-8", FormatType.JSON);
            try
            {
                VideoAsyncScanResultsResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

视频截帧同步检测

接口	描述	支持的地域

接口	描述	支持的地域
VideoSyncScanRequest	提交视频同步检测任务，同步检测视频中的风险内容。 **说明** 同步检测只支持传递视频帧序列，不支持检测视频文件，推荐使用异步检测接口。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

以下示例代码中使用帧序列方式提交待检测的视频。

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            VideoSyncScanRequest request = new VideoSyncScanRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            Dictionary<string, object> frame1 = new Dictionary<string, object>();
            frame1.Add("offset", "0");
            frame1.Add("url", "您的视频截帧链接地址1");
            Dictionary<string, object> frame2 = new Dictionary<string, object>();
            frame2.Add("offset", "5");
            frame2.Add("url", "您的视频截帧链接地址2");
            Dictionary<string, object> frame3 = new Dictionary<string, object>();
            frame3.Add("offset", "10");
            frame3.Add("url", "您的视频截帧链接地址3");
            Dictionary<string, object> task1 = new Dictionary<string, object>();
            task1.Add("dataId", "检测数据ID");
            task1.Add("frames", new List<Dictionary<string, object>> { frame1, frame2, fram
e3 });
            Dictionary<string, object> httpBody = new Dictionary<string, object>();
            // scenes：检测场景，支持指定多个场景。
```

```
            httpBody.Add("scenes", new List<string> { "porn", "terrorism" });
            httpBody.Add("bizType", "业务场景");
            httpBody.Add("tasks", new List<Dictionary<string, object>> { task1 });
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(httpBody)), "utf-8", FormatType.JSON);
            try
            {
                VideoSyncScanResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

视频检测结果反馈

如果您认为视频检测结果与您的预期不符，可以通过视频检测结果反馈接口，对检测结果进行纠正（系统会根据您反馈的结果，将视频截帧添加到相似图片的黑名单库或者白名单库）。当您再次提交相似的内容进行检测时，以您反馈的label返回结果。

关于接口的说明，请参见检测结果反馈。

接口	描述	支持的Region
VideoFeedbackRequest	提交视频检测结果的反馈，以人工反馈的检测结果纠正算法检测结果。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
```

```
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            VideoFeedbackRequest request = new VideoFeedbackRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            Dictionary<string, object> frame1 = new Dictionary<string, object>();
            frame1.Add("offset", "0");
            frame1.Add("url", "您的视频截帧链接地址1");
            Dictionary<string, object> frame2 = new Dictionary<string, object>();
            frame2.Add("offset", "0");
            frame2.Add("url", "您的视频截帧链接地址2");
            List<Dictionary<string, object>> frames = new List<Dictionary<string, object>>
{ frame1, frame2 };
            // scenes：检测场景，支持指定多个场景。
            // suggestion：期望的检测结果，pass：正常；block：违规。
            Dictionary<string, object> httpBody = new Dictionary<string, object>();
            httpBody.Add("scenes", new List<string> { "porn", "terrorism" });
            httpBody.Add("suggestion", "block");
            httpBody.Add("taskId", "视频审核任务ID");
            httpBody.Add("dataId", "检测数据ID");
            httpBody.Add("url", "视频链接地址");
            httpBody.Add("frames", frames);
            httpBody.Add("note", "备注信息");
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(httpBody)), "utf-8", FormatType.JSON);
            try
            {
                VideoFeedbackResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

6.3.3. 文本反垃圾检测

本文介绍了如何使用.NET SDK文本反垃圾接口，对文本内容进行色情、暴恐、涉政等风险进行识别。

功能描述

文本反垃圾接口目前仅支持同步检测。关于参数的详细说明，请参见文本同步检测API文档。

一次请求可以检测多条文本，也可以检测单条文本。按实际检测的文本条数进行计费，请参见计费方式概述。

前提条件

已安装.NET依赖。关于安装.NET依赖的具体操作，请参见安装.NET依赖。

> **说明** 请一定按照安装.NET依赖页面中的版本安装，否则会导致调用失败。

文本内容检测

文本垃圾检测支持自定义关键词，例如添加一些竞品关键词等。如果被检测的文本中包含您添加的关键词，算法会返回您block。

您可以前往内容安全控制台添加关键词，也可以通过API接口添加关键词。

接口	描述	支持的Region
TextScanRequest	提交文本反垃圾检测任务，检测场景参数请传递antispam（scenes=antispam）。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            TextScanRequest request = new TextScanRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            Dictionary<string, object> task1 = new Dictionary<string, object>();
            task1.Add("content", "待检测文本内容");
            Dictionary<string, object> httpBody = new Dictionary<string, object>();
            // 检测场景。文本垃圾检测请传递antispam。
            httpBody.Add("scenes", new List<string> { "antispam" });
            httpBody.Add("bizType", "default");
            httpBody.Add("tasks", new List<Dictionary<string, object>> { task1 });
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(httpBody)), "utf-8", FormatType.JSON);
            try
            {
                TextScanResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

文本反垃圾结果反馈

如果您认为文本检测的结果与您的期望不符，您可以通过文本反垃圾结果反馈接口纠正算法的检测结果。

系统会将您反馈的结果添加到对应的检测文本库，在您下次提交相似的内容进行检测时，以您通过反馈接口校正后的结果作为检测结果。

接口	描述	支持的地域
TextFeedbackRequest	提交文本反垃圾检测结果的反馈，以人工反馈的检测结果纠正算法检测结果。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            TextFeedbackRequest request = new TextFeedbackRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            // label：反馈的分类，与具体的scene对应。
            Dictionary<string, object> httpBody = new Dictionary<string, object>();
            httpBody.Add("taskId", "文本审核任务ID");
            httpBody.Add("label", "spam");
            httpBody.Add("content", "文本内容");
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(httpBody)), "utf-8", FormatType.JSON);
            try
            {
                TextFeedbackResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

6.3.4. 语音反垃圾检测

本文介绍了如何使用.NET SDK语音反垃圾接口，检测实时语音流或语音文件中的垃圾内容。

功能描述

语音流检测和语音文件检测均为异步检测，检测结果需要您以轮询或者回调的方式获取。关于调用请求中的检测场景参数scenes，返回结果中的分类参数label，以及操作建议参数suggestion的说明，请参见语音异步检测API文档。

语音检测按照检测的语音文件、语音流的时间长度进行计费，计费粒度为分钟，每天累计检测总时长进行计量统计，每天检测总时长不足一分钟的按照一分钟进行计费。

前提条件

已安装.NET依赖。关于安装.NET依赖的具体操作，请参见安装.NET依赖。

> **说明** 请一定按照安装.NET依赖页面中的版本安装，否则会导致调用失败。

提交语音异步检测任务

接口	描述	支持的地域
VoiceAsyncScanRequest	异步检测语音流或语音文件中是否包含违规内容。语音流格式支持： • RTMP • HTTP • FLV	**cn-shanghai**：华东2（上海）

示例代码

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            VoiceAsyncScanRequest request = new VoiceAsyncScanRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
```

```
            Dictionary<string, object> task1 = new Dictionary<string, object>();
            task1.Add("dataId", "检测数据ID");
            task1.Add("url", "待检测音频链接地址");
            // scenes：检测场景，唯一取值：antispam。
            // callback、seed用于回调通知，可选参数。
            Dictionary<string, object> httpBody = new Dictionary<string, object>();
            httpBody.Add("scenes", new List<string> { "antispam" });
            httpBody.Add("bizType", "业务场景");
            httpBody.Add("callback", "回调地址");
            httpBody.Add("seed", "随机字符串");
            httpBody.Add("tasks", new List<Dictionary<string, object>> { task1 });
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(httpBody)), "utf-8", FormatType.JSON);
            try
            {
                VoiceAsyncScanResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

查询语音异步检测结果

接口	描述	支持的地域
VoiceAsyncScanResultsRequest	查询异步语音检测结果。	`cn-shanghai`：华东2（上海）

示例代码

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            VoiceAsyncScanResultsRequest request = new VoiceAsyncScanResultsRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            List<string> taskIdList = new List<string> { "音频检测任务ID" };
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(taskIdList)), "utf-8", FormatType.JSON);
            try
            {
                VoiceAsyncScanResultsResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

短语音同步检测

接口	描述	支持的地域
VoicesyncscanRequest	查询短语音检测结果。	`cn-shanghai`：华东2（上海）

示例代码

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            VoiceSyncScanRequest request = new VoiceSyncScanRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            Dictionary<string, object> task1 = new Dictionary<string, object>();
            task1.Add("dataId", "检测数据ID");
            task1.Add("url", "待检测音频链接地址");
            Dictionary<string, object> httpBody = new Dictionary<string, object>();
            // scenes：检测场景，支持指定多个场景。
            httpBody.Add("scenes", new List<string> { "porn", "terrorism" });
            httpBody.Add("bizType", "业务场景");
            httpBody.Add("tasks", new List<Dictionary<string, object>> { task1 });
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(httpBody)), "utf-8", FormatType.JSON);
            try
            {
                VoiceSyncScanResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

6.4. 人工审核

6.4.1. 图片人工审核

本文介绍了如何使用.NET SDK图片人工审核接口，进行人工审核。

功能描述

如果您认为图片检测结果与您的预期不符，可以进行人工审核。关于参数的详细信息，请参见图片人工审核API文档。

关于该API的接入地址，请参见接入地址（Endpoint）。

前提条件

已安装.NET依赖。关于安装.NET依赖的具体操作，请参见安装.NET依赖。

> **说明** 请一定按照安装.NET依赖页面中的版本安装，否则会导致调用失败。

提交图片人工审核任务

示例代码

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            ImageAsyncManualScanRequest request = new ImageAsyncManualScanRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            Dictionary<string, object> task1 = new Dictionary<string, object>();
            task1.Add("dataId", "检测数据ID");
            task1.Add("url", "图片链接地址");
            // callback、seed用于回调通知，可选参数。
            Dictionary<string, object> httpBody = new Dictionary<string, object>();
            httpBody.Add("bizType", "业务场景");
            httpBody.Add("callback", "回调地址");
            httpBody.Add("seed", "随机字符串");
            httpBody.Add("tasks", new List<Dictionary<string, object>> { task1 });
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(httpBody)), "utf-8", FormatType.JSON);
            try
            {
                ImageAsyncManualScanResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

查询图片人工审核任务结果

示例代码

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            ImageAsyncManualScanResultsRequest request = new ImageAsyncManualScanResultsReq
uest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            List<string> tasks = new List<string>();
            tasks.Add("图片人工审核任务ID");
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(tasks)), "utf-8", FormatType.JSON);
            try
            {
                ImageAsyncManualScanResultsResponse response = client.GetAcsResponse(reques
t);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

6.4.2. 视频人工审核

本文介绍了如何使用.NET SDK视频人工审核接口，进行人工审核。

功能描述

如果您认为视频检测结果与您的预期不符，可以进行人工审核。关于参数的详细信息，请参见视频人工审核API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装.NET依赖。关于安装.NET依赖的具体操作，请参见安装.NET依赖。

> **说明** 请一定按照安装.NET依赖页面中的版本安装，否则会导致调用失败。

提交视频人工审核任务

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            VideoAsyncManualScanRequest request = new VideoAsyncManualScanRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            Dictionary<string, object> task1 = new Dictionary<string, object>();
            task1.Add("dataId", "检测数据ID");
            task1.Add("url", "视频链接地址");
            // callback、seed用于回调通知，可选参数。
            Dictionary<string, object> httpBody = new Dictionary<string, object>();
            httpBody.Add("bizType", "业务场景");
            httpBody.Add("callback", "回调地址");
            httpBody.Add("seed", "随机字符串");
            httpBody.Add("tasks", new List<Dictionary<string, object>> { task1 });
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(httpBody)), "utf-8", FormatType.JSON);
            try
            {
                VideoAsyncManualScanResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

查询视频人工审核任务结果

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            VideoAsyncManualScanResultsRequest request = new VideoAsyncManualScanResultsReq
uest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            List<string> tasks = new List<string>();
            tasks.Add("视频人工审核任务ID");
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(tasks)), "utf-8", FormatType.JSON);
            try
            {
                VideoAsyncManualScanResultsResponse response = client.GetAcsResponse(reques
t);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

6.4.3. 文本人工审核

本文介绍了如何使用.NET SDK文本人工审核接口，进行人工审核。

功能描述

如果您认为文本检测结果与您的预期不符，可以进行人工审核。关于参数的详细信息，请参见文本人工审核API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装.NET依赖。关于安装.NET依赖的具体操作，请参见安装.NET依赖。

> **说明** 请一定按照安装.NET依赖页面中的版本安装，否则会导致调用失败。

提交视频人工审核任务

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
       {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            TextAsyncManualScanRequest request = new TextAsyncManualScanRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            Dictionary<string, object> task1 = new Dictionary<string, object>();
            task1.Add("dataId", "检测数据ID");
            task1.Add("url", "待检测文本");
            // callback、seed用于回调通知，可选参数。
            Dictionary<string, object> httpBody = new Dictionary<string, object>();
            httpBody.Add("bizType", "业务场景");
            httpBody.Add("callback", "回调地址");
            httpBody.Add("seed", "随机字符串");
            httpBody.Add("tasks", new List<Dictionary<string, object>> { task1 });
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(httpBody)), "utf-8", FormatType.JSON);
            try
            {
                TextAsyncManualScanResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

查询文本人工审核任务结果

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            TextAsyncManualScanResultsRequest request = new TextAsyncManualScanResultsReque
st();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            List<string> tasks = new List<string>();
            tasks.Add("文本人工审核任务ID");
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(tasks)), "utf-8", FormatType.JSON);
            try
            {
                TextAsyncManualScanResultsResponse response = client.GetAcsResponse(request
);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

6.4.4. 语音人工审核

本文介绍了如何使用.NET SDK语音人工审核接口，进行人工审核。

功能描述

如果您认为语音检测结果与您的预期不符，可以进行人工审核。关于参数的详细信息，请参见语音人工审核API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装.NET依赖。关于安装.NET依赖的具体操作，请参见安装.NET依赖。

> **说明** 请一定按照安装.NET依赖页面中的版本安装，否则会导致调用失败。

提交语音人工审核任务

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            VoiceAsyncManualScanRequest request = new VoiceAsyncManualScanRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            Dictionary<string, object> task1 = new Dictionary<string, object>();
            task1.Add("dataId", "检测数据ID");
            task1.Add("url", "待检测音频链接地址");
            // callback、seed用于回调通知，可选参数。
            Dictionary<string, object> httpBody = new Dictionary<string, object>();
            httpBody.Add("bizType", "业务场景");
            httpBody.Add("callback", "回调地址");
            httpBody.Add("seed", "随机字符串");
            httpBody.Add("tasks", new List<Dictionary<string, object>> { task1 });
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(httpBody)), "utf-8", FormatType.JSON);
            try
            {
                VoiceAsyncManualScanResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

查询语音人工审核任务结果

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            VoiceAsyncManualScanResultsRequest request = new VoiceAsyncManualScanResultsReq
uest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            List<string> tasks = new List<string>();
            tasks.Add("音频人工审核任务ID");
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(tasks)), "utf-8", FormatType.JSON);
            try
            {
                VoiceAsyncManualScanResultsResponse response = client.GetAcsResponse(reques
t);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

6.5. 图片OCR识别

本文介绍了如何使用.NET SDK图片OCR接口，识别图片中的文字或卡证信息。

功能描述

通用OCR除了能够识别普通图片中的文字，还能识别结构化卡证上的文字。关于参数的详细说明，请参见图片OCR检测API文档。

前提条件

已安装.NET依赖。关于安装.NET依赖的具体操作，请参见安装.NET依赖。

> **说明** 请一定按照安装.NET依赖页面中的版本安装，否则会导致调用失败。

提交图片同步检测任务

接口	描述	支持的Region
ImageSyncScanRequest	提交图片OCR同步识别任务，对图片中的文字进行识别（scene=ocr）。	• *cn-shanghai* • *cn-beijing* • *cn-shenzhen* • *ap-southeast-1*

示例代码

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            ImageSyncScanRequest request = new ImageSyncScanRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            Dictionary<string, object> task1 = new Dictionary<string, object>();
            task1.Add("dataId", "检测数据ID");
            //task1.Add("url", "待检测图片链接地址");
            task1.Add("url", "http://example.com/xx.jpg");
```

```
            // 示例：身份证正面识别。
            Dictionary<string, object> cardExtras = new Dictionary<string, object>();
            cardExtras.Add("card", "id-card-front");
            Dictionary<string, object> httpBody = new Dictionary<string, object>();
            // scenes：检测场景。
            httpBody.Add("scenes", new List<string> { "ocr" });
            httpBody.Add("bizType", "业务场景");
            httpBody.Add("extras", cardExtras);
            httpBody.Add("tasks", new List<Dictionary<string, object>> { task1 });
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(httpBody)), "utf-8", FormatType.JSON);
            try
            {
                ImageSyncScanResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

6.6. 人脸识别

6.6.1. 人脸比对

本文介绍了如何使用.NET SDK人脸比对接口，对指定的人脸图片进行属性检测。人脸属性检测仅支持以图片作为检测媒介。

功能描述

人脸比对支持同步检测和异步检测两种方式。

- 同步检测实时返回检测结果。关于参数的详细信息，请参见同步检测API文档。
- 异步检测需要您轮询结果或者通过callback回调通知获取检测结果。关于参数的详细信息，请参见异步检测API文档。

前提条件

已安装.NET依赖。关于安装.NET依赖的具体操作，请参见安装.NET依赖。

> **说明** 请一定按照安装.NET依赖页面中的版本安装，否则会导致调用失败。

提交人脸比对任务

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            ImageSyncScanRequest request = new ImageSyncScanRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            // 设置待比对人脸图片链接地址2。
            Dictionary<string, object> extras = new Dictionary<string, object>();
            extras.Add("faceUrl", "待比对人脸图片链接地址2");
            // 设置待比对人脸图片链接地址1。
            Dictionary<string, object> task1 = new Dictionary<string, object>();
            task1.Add("dataId", "检测数据ID");
            task1.Add("url", "待比对人脸图片链接地址1");
            task1.Add("extras", extras);
            // scenes：检测场景，指定图片检测的应用场景，取值：sface-1。
            Dictionary<string, object> httpBody = new Dictionary<string, object>();
            httpBody.Add("scenes", new List<string> { "sface-1" });
            httpBody.Add("bizType", "业务场景");
            httpBody.Add("tasks", new List<Dictionary<string, object>> { task1 });
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(httpBody)), "utf-8", FormatType.JSON);
            try
            {
                ImageSyncScanResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
```

```
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

6.6.2. 人脸属性检测

本文介绍了如何使用.NET SDK人脸属性检测接口，对指定的人脸图片（公网图片）进行属性检测。人脸属性检测仅支持以图片作为检测媒介。

功能描述

人脸属性检测能够识别图片中的人脸属性信息，包括人脸模糊度、人脸角度、人脸位置、微笑程度、是否戴眼镜、是否戴口罩、是否戴帽子、是否有胡子、是否有刘海、头发类型等。关于参数的详细说明，请参见人脸属性检测API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装.NET依赖。关于安装.NET依赖的具体操作，请参见安装.NET依赖。

> **说明** 请一定按照安装.NET依赖页面中的版本安装，否则会导致调用失败。

提交人脸属性检测任务

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            DetectFaceRequest request = new DetectFaceRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            Dictionary<string, object> httpBody = new Dictionary<string, object>();
            httpBody.Add("url", "待检测人脸图片链接地址");
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(httpBody)), "utf-8", FormatType.JSON);
            try
            {
                DetectFaceResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

6.6.3. 自定义人脸检索

6.6.3.1. 自定义人脸检索

本文介绍了如何使用.NET SDK自定义人脸检索接口，对指定的人脸图片进行自定义人脸检索。

功能描述

自定义人脸检索根据您输入的待识别人脸图片（face），在指定人脸库（group）中查找并返回最相似的Top 5个体（person），返回的Top 5个体按照相似度从大到小排序。

检索人脸时，必须指定要检索的分组信息（最多指定一个分组）。关于参数的详细说明，请参见自定义人脸检索API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装.NET依赖。关于安装.NET依赖的具体操作，请参见安装.NET依赖。

> **说明** 请一定按照安装.NET依赖页面中的版本安装，否则会导致调用失败。

提交人脸图片检索任务

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            ImageSyncScanRequest request = new ImageSyncScanRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            Dictionary<string, object> extras = new Dictionary<string, object>();
            extras.Add("groupId", "个体组ID");
            Dictionary<string, object> task1 = new Dictionary<string, object>();
            task1.Add("dataId", "检测数据ID");
            task1.Add("url", "待检测人脸图片链接地址");
            task1.Add("extras", extras);
            // scenes：检测场景，取值：sface-n，表示自定义人脸检索。
            Dictionary<string, object> httpBody = new Dictionary<string, object>();
            httpBody.Add("scenes", new List<string> { "sface-n" });
            httpBody.Add("bizType", "业务场景");
```

```
            httpBody.Add("tasks", new List<Dictionary<string, object>> { task1 });
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(httpBody)), "utf-8", FormatType.JSON);
            try
            {
                ImageSyncScanResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

6.6.3.2. 新建个体

本文介绍了如何使用.NET SDK新建个体信息。

功能描述

新建个体时，必须指定个体要加入的分组。关于参数的详细说明，请参见新建个体API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装.NET依赖。关于安装.NET依赖的具体操作，请参见安装.NET依赖。

> **说明** 请一定按照安装.NET依赖页面中的版本安装，否则会导致调用失败。

新建个体任务

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
```

```
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            AddPersonRequest request = new AddPersonRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            /**
            * personId: 用户自定义个体ID，必填。
            * groupIds: 用户自定义组ID列表，必填。
            * name: 用户名称，非必填。
            * note: 备注信息，非必填。
            */
            Dictionary<string, object> httpBody = new Dictionary<string, object>();
            httpBody.Add("personId", "个体ID");
            httpBody.Add("groupIds", new List<string> { "组ID_1", "组ID_2" });
            httpBody.Add("name", "名称");
            httpBody.Add("note", "备注信息");
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(httpBody)), "utf-8", FormatType.JSON);
            try
            {
                AddPersonResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

6.6.3.3. 设置个体

本文介绍了如何使用.NET SDK设置个体信息。

功能描述

设置个体信息用于给个体添加备注信息，属于可选步骤。如果您设置了个体信息，则在自定义检索的返回结果中会包含该信息。关于参数的详细说明，请参见设置个体API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装.NET依赖。关于安装.NET依赖的具体操作，请参见安装.NET依赖。

> **说明** 请一定按照安装.NET依赖页面中的版本安装，否则会导致调用失败。

提交设置个体任务

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
         {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            SetPersonRequest request = new SetPersonRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            /**
            * personId: 用户自定义个体ID，必填。
            * name: 用户名称，非必填。
            * note: 备注信息，非必填。
            */
            Dictionary<string, object> httpBody = new Dictionary<string, object>();
            httpBody.Add("personId", "个体ID");
            httpBody.Add("name", "名称");
            httpBody.Add("note", "备注信息");
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(httpBody)), "utf-8", FormatType.JSON);
            try
            {
                SetPersonResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

6.6.3.4. 查询个体信息

本文介绍了如何使用.NET SDK查询个体信息。

功能描述

关于参数的详细说明，请参见查询个体信息API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装.NET依赖。关于安装.NET依赖的具体操作，请参见安装.NET依赖。

> **说明** 请一定按照安装.NET依赖页面中的版本安装，否则会导致调用失败。

查询个体信息任务

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            GetPersonRequest request = new GetPersonRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            /**
            * personId: 用户自定义个体ID，必填。
            */
            Dictionary<string, object> httpBody = new Dictionary<string, object>();
            httpBody.Add("personId", "个体ID");
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(httpBody)), "utf-8", FormatType.JSON);
            try
            {
                GetPersonResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

6.6.3.5. 查询人脸信息

本文介绍了如何使用.NET SDK查询人脸信息。

功能描述

根据个体ID和人脸ID查询当前个体的人脸ID以及人脸图片路径等信息。关于查询人脸的具体参数说明，请参见查询人脸图片API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装.NET依赖。关于安装.NET依赖的具体操作，请参见安装.NET依赖。

> **说明** 请一定按照安装.NET依赖页面中的版本安装，否则会导致调用失败。

查询人脸信息任务

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            GetFacesRequest request = new GetFacesRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            /**
            * personId: 用户自定义个体ID，必填。
            * faceIds: 人脸ID集合，必填。
            */
            Dictionary<string, object> httpBody = new Dictionary<string, object>();
            httpBody.Add("personId", "个体ID");
            httpBody.Add("faceIds", new List<string> { "人脸ID_1", "人脸ID_2" });
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(httpBody)), "utf-8", FormatType.JSON);
            try
            {
                GetFacesResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

6.6.3.6. 删除个体

本文介绍了如何使用.NET SDK删除个体信息。

功能描述

删除个体时，个体对应的图片以及信息均会被删除。关于参数的详细说明，请参见删除个体API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装.NET依赖。关于安装.NET依赖的具体操作，请参见安装.NET依赖。

> **说明** 请一定按照安装.NET依赖页面中的版本安装，否则会导致调用失败。

删除个体任务

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            DeletePersonRequest request = new DeletePersonRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            /**
            * personId: 用户自定义个体ID，必填。
            */
            Dictionary<string, object> httpBody = new Dictionary<string, object>();
            httpBody.Add("personId", "个体ID");
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(httpBody)), "utf-8", FormatType.JSON);
            try
            {
                DeletePersonResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

6.6.3.7. 新增人脸

本文介绍了如何使用.NET SDK新增个体的人脸图片。

功能描述

为个体新增一组人脸图片，一个个体最多允许包含20张人脸图片。关于参数的说明，请参见新增人脸API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装.NET依赖。关于安装.NET依赖的具体操作，请参见安装.NET依赖。

> **说明** 请一定按照安装.NET依赖页面中的版本安装，否则会导致调用失败。

新增人脸图片任务

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            AddFacesRequest request = new AddFacesRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            /**
            * personId：用户自定义个体ID，必填。
            * urls：人脸图片URL。
            */
            Dictionary<string, object> httpBody = new Dictionary<string, object>();
            httpBody.Add("personId", "个体ID");
            httpBody.Add("urls", new List<string> { "人脸图片链接地址1", "人脸图片链接地址2" })
;
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(httpBody)), "utf-8", FormatType.JSON);
            try
            {
                AddFacesResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

6.6.3.8. 删除人脸

本文介绍了如何使用.NET SDK，删除无用的个体人脸图片。

功能描述

删除人脸图片时，必须指定要删除的图片ID以及对应的个体ID信息。关于参数的详细说明，请参见删除人脸API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装.NET依赖。关于安装.NET依赖的具体操作，请参见安装.NET依赖。

> **说明** 请一定按照安装.NET依赖页面中的版本安装，否则会导致调用失败。

提交删除人脸任务

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            DeleteFacesRequest request = new DeleteFacesRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            /**
            * personId：用户自定义个体ID，必填。
            * faceIds：已添加的人脸ID。
            */
            Dictionary<string, object> httpBody = new Dictionary<string, object>();
            httpBody.Add("personId", "个体ID");
            httpBody.Add("faceIds", new List<string> { "人脸ID_1", "人脸ID_2" });
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(httpBody)), "utf-8", FormatType.JSON);
            try
            {
                DeleteFacesResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

6.6.3.9. 查询组列表

本文介绍了如何使用.NET SDK查询所有人脸分组ID。

功能描述

查询所有的个体组ID。关于参数的详细说明，请参见查询组列表API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装.NET依赖。关于安装.NET依赖的具体操作，请参见安装.NET依赖。

> **说明** 请一定按照安装.NET依赖页面中的版本安装，否则会导致调用失败。

查询所有分组ID任务

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            GetGroupsRequest request = new GetGroupsRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            try
            {
                GetGroupsResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

6.6.3.10. 查询个体列表

本文介绍了如何使用.NET SDK查询个体列表信息。

功能描述

每个个体必须属于一个分组，在调用该接口时指定某个分组，则返回该分组下的所有个体ID列表。关于参数的详细说明，请参见查询个体列表API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装.NET依赖。关于安装.NET依赖的具体操作，请参见安装.NET依赖。

> **说明** 请一定按照安装.NET依赖页面中的版本安装，否则会导致调用失败。

查询个体列表任务

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            GetPersonsRequest request = new GetPersonsRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            /**
            * groupId：用户自定义组ID，必填。
            */
            Dictionary<string, object> httpBody = new Dictionary<string, object>();
            httpBody.Add("groupId", "组ID");
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(httpBody)), "utf-8", FormatType.JSON);
            try
            {
                GetPersonsResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

6.6.3.11. 添加个体到分组

本文介绍了如何使用.NET SDK添加个体到指定分组。

功能描述

将一个个体添加到一个或者多个个体组。关于参数的详细说明，请参见添加个体到分组API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装.NET依赖。关于安装.NET依赖的具体操作，请参见安装.NET依赖。

> **说明** 请一定按照安装.NET依赖页面中的版本安装，否则会导致调用失败。

添加个体到分组任务

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            AddGroupsRequest request = new AddGroupsRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            /**
            * personId：用户自定义个体ID，必填。
            * groupIds：用户自定义组ID列表，必填。
            */
            Dictionary<string, object> httpBody = new Dictionary<string, object>();
            httpBody.Add("personId", "个体ID");
            httpBody.Add("groupIds", new List<string> { "组ID_1", "组ID_2" });
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(httpBody)), "utf-8", FormatType.JSON);
            try
            {
                AddGroupsResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

6.6.3.12. 删除分组中个体

本文介绍了如何使用.NET SDK从指定分组中删除个体信息。

功能描述

删除分组中个体不会删除个体对应的信息以及图片，只是解除个体与该分组的绑定关系。关于参数的详细说明，请参见删除分组中的个体API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装.NET依赖。关于安装.NET依赖的具体操作，请参见安装.NET依赖。

> **说明** 请一定按照安装.NET依赖页面中的版本安装，否则会导致调用失败。

从指定分组删除个体任务

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            DeleteGroupsRequest request = new DeleteGroupsRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            /**
            * personId：用户自定义个体ID，必填。
            * groupIds：用户自定义组ID列表，必填。
            */
            Dictionary<string, object> httpBody = new Dictionary<string, object>();
            httpBody.Add("personId", "个体ID");
            httpBody.Add("groupIds", new List<string> { "组ID_1", "组ID_2" });
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(httpBody)), "utf-8", FormatType.JSON);
            try
            {
                DeleteGroupsResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

6.7. 其他

6.7.1. 相似图检索

6.7.1.1. 创建相似图样本库

本文介绍了如何使用.NET SDK，创建相似图样本库。

功能描述

通过该接口创建相似图样本图库。关于参数的详细说明，请参见创建图库API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装.NET依赖。关于安装.NET依赖的具体操作，请参见安装.NET依赖。

> **说明** 请一定按照安装.NET依赖页面中的版本安装，否则会导致调用失败。

创建相似图样本任务

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            AddSimilarityLibraryRequest request = new AddSimilarityLibraryRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            Dictionary<string, object> httpBody = new Dictionary<string, object>();
            httpBody.Add("name", "相似图样本图库名称");
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(httpBody)), "utf-8", FormatType.JSON);
            try
            {
                AddSimilarityLibraryResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

6.7.1.2. 相似图检索

本文介绍了如何使用.NET SDK，对指定的图片进行相似图检索。

功能描述

相似图检索帮助您从图库中检索出与给定的图片相似的若干张图片。关于参数的详细说明，请参见相似图检索API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装.NET依赖。关于安装.NET依赖的具体操作，请参见安装.NET依赖。

> **说明** 请一定按照安装.NET依赖页面中的版本安装，否则会导致调用失败。

提交相似图检索任务

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            ImageSyncScanRequest request = new ImageSyncScanRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            Dictionary<string, object> task1 = new Dictionary<string, object>();
            task1.Add("dataId", "检测数据ID");
            task1.Add("url", "待检测图片链接地址");
            task1.Add("similarityLibraries", new List<string> { "相似图样本图库名称_1", "相似图样本图库名称_2" });
            Dictionary<string, object> httpBody = new Dictionary<string, object>();
            /**
            * 设置要检测的场景，计费是按照该处传递的场景进行。
            * similarity：表示进行相似图检索。
            */
            httpBody.Add("scenes", new List<string> { "similarity" });
            httpBody.Add("bizType", "业务场景");
            httpBody.Add("tasks", new List<Dictionary<string, object>> { task1 });
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeObject(httpBody)), "utf-8", FormatType.JSON);
            try
```

```
            {
                ImageSyncScanResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

6.7.1.3. 增加相似图样本

本文介绍了如何使用.NET SDK，将指定的相似图图片添加到对应图库。

功能描述

添加相似图样本时，您可以为图片设置标签。如果样本图片在检索时被命中，其标签信息也会被返回。关于参数的详细说明，请参见增加样本图片API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装.NET依赖。关于安装.NET依赖的具体操作，请参见安装.NET依赖。

> **说明** 请一定按照安装.NET依赖页面中的版本安装，否则会导致调用失败。

增加相似图样本任务

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            AddSimilarityImageRequest request = new AddSimilarityImageRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            Dictionary<string, object> task = new Dictionary<string, object>();
            task.Add("dataId", "检测数据ID");
            task.Add("url", "样本图片链接");
            task.Add("tag", new List<string> { "自定义标签1", "自定义标签2", "自定义标签3" });
            Dictionary<string, object> httpBody = new Dictionary<string, object>();
            httpBody.Add("name", "相似图样本图库名称");
            httpBody.Add("tasks", new List<Dictionary<string, object>> { task });
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(httpBody)), "utf-8", FormatType.JSON);
            try
            {
                AddSimilarityImageResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

6.7.1.4. 查询相似图库列表

本文介绍了如何使用.NET SDK，分页查询相似图库及其元素信息。

功能描述

调用该接口分页查询所有相似图库及其元素信息。关于参数的详细说明，请参见查询相似图库列表API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装.NET依赖。关于安装.NET依赖的具体操作，请参见安装.NET依赖。

> **说明** 请一定按照安装.NET依赖页面中的版本安装，否则会导致调用失败。

查询相似图库列表任务

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            ListSimilarityLibrariesRequest request = new ListSimilarityLibrariesRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            Dictionary<string, object> httpBody = new Dictionary<string, object>();
            httpBody.Add("pageSize", "5");
            httpBody.Add("currentPage", "1");
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(httpBody)), "utf-8", FormatType.JSON);
            try
            {
                ListSimilarityLibrariesResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

6.7.1.5. 查询指定相似图库信息

本文介绍了如何使用.NET SDK，查询指定相似图库信息。

功能描述

调用该接口查询指定相似图库信息。关于参数的详细说明，请参见查询指定相似图库API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装.NET依赖。关于安装.NET依赖的具体操作，请参见安装.NET依赖。

> **说明** 请一定按照安装.NET依赖页面中的版本安装，否则会导致调用失败。

查询指定相似图库任务

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            GetSimilarityLibraryRequest request = new GetSimilarityLibraryRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            Dictionary<string, object> httpBody = new Dictionary<string, object>();
            httpBody.Add("name", "相似图样本图库名称");
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(httpBody)), "utf-8", FormatType.JSON);
            try
            {
                GetSimilarityLibraryResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

6.7.1.6. 查询相似图样本图片列表

本文介绍了如何使用.NET SDK，分页查询指定相似图库下的所有样本图片信息。

功能描述

调用该接口分页查询指定相似图库下的所有样本图片信息。关于参数的详细说明，请参见查询样本图片列表API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装.NET依赖。关于安装.NET依赖的具体操作，请参见安装.NET依赖。

> **说明** 请一定按照安装.NET依赖页面中的版本安装，否则会导致调用失败。

查询相似图样本图片列表任务

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            ListSimilarityImagesRequest request = new ListSimilarityImagesRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            // pageSize：每页大小，取值范围：(0,50]；currentPage：当前页码，取值范围：(0,50]
            Dictionary<string, object> httpBody = new Dictionary<string, object>();
            httpBody.Add("library", "相似图样本图库名称");
            httpBody.Add("pageSize", "5");
            httpBody.Add("currentPage", "1");
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(httpBody)), "utf-8", FormatType.JSON);
            try
            {
                ListSimilarityImagesResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

6.7.1.7. 查询相似图样本图片详情

本文介绍了如何使用.NET SDK，查询相似图样本图片详情。

功能描述

调用该接口查询相似图样本图片详情。关于参数的详细说明，请参见查询样本图库详情API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装.NET依赖。关于安装.NET依赖的具体操作，请参见安装.NET依赖。

> **说明** 请一定按照安装.NET依赖页面中的版本安装，否则会导致调用失败。

查询相似图样本图片详情任务

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            GetSimilarityImageRequest request = new GetSimilarityImageRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            // pageSize：每页大小，取值范围：(0,50]；currentPage：当前页码，取值范围：(0,50]。
            Dictionary<string, object> httpBody = new Dictionary<string, object>();
            httpBody.Add("library", "相似图样本图库名称");
            httpBody.Add("dataId", "样本图片ID");
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(httpBody)), "utf-8", FormatType.JSON);
            try
            {
                GetSimilarityImageResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

6.7.1.8. 移除相似图样本

本文介绍了如何使用.NET SDK，将相似图样本从图库中移除。

功能描述

移除操作在1分钟之内生效。一次最多允许移除100张样本图片。关于参数的详细说明，请参见删除样本图片API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装.NET依赖。关于安装.NET依赖的具体操作，请参见安装.NET依赖。

> **说明** 请一定按照安装.NET依赖页面中的版本安装，否则会导致调用失败。

移除相似图样本任务

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            DeleteSimilarityImageRequest request = new DeleteSimilarityImageRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            Dictionary<string, object> httpBody = new Dictionary<string, object>();
            httpBody.Add("dataIds", new List<string> { "样本图片ID_1", "样本图片ID_2" });
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(httpBody)), "utf-8", FormatType.JSON);
            try
            {
                DeleteSimilarityImageResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

6.7.1.9. 删除相似图库

本文介绍了如何使用.NET SDK，删除指定相似图库。

功能描述

只有当相似图库中无图片样本时，才允许被删除。关于参数的详细说明，请参见删除图库API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装.NET依赖。关于安装.NET依赖的具体操作，请参见安装.NET依赖。

> **说明** 请一定按照安装.NET依赖页面中的版本安装，否则会导致调用失败。

删除相似图库任务

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20180509;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
         {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            DeleteSimilarityLibraryRequest request = new DeleteSimilarityLibraryRequest();
            request.AcceptFormat = FormatType.JSON;
            request.ContentType = FormatType.JSON;
            request.Method = MethodType.POST;
            request.Encoding = "UTF-8";
            // pageSize：每页大小，取值范围：(0,50]；currentPage：当前页码，取值范围：(0,50]。
            Dictionary<string, object> httpBody = new Dictionary<string, object>();
            httpBody.Add("name", "相似图样本图库名称");
            request.SetContent(System.Text.Encoding.Default.GetBytes(JsonConvert.SerializeO
bject(httpBody)), "utf-8", FormatType.JSON);
            try
            {
                DeleteSimilarityLibraryResponse response = client.GetAcsResponse(request);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

6.7.2. 自定义文本库

本文介绍了如何使用.NET SDK管理自定义文本库，以满足文本反垃圾检测场景的个性化需求。

功能描述

根据文本类型的不同，文本库分为关键词文本库和相似文本文本库；根据管控目的不同，文本库分为白名单、黑名单、疑似名单。关于参数的详细信息，请参见自定义文本库API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装.NET依赖。关于安装.NET依赖的具体操作，请参见安装.NET依赖。

> **说明** 请一定按照安装.NET依赖页面中的版本安装，否则会导致调用失败。

查询文本库列表

- 查询关键词文本库列表

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20170823;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            DescribeKeywordLibRequest request = new DescribeKeywordLibRequest();
            request.ServiceModule = "open_api";
            try
            {
                DescribeKeywordLibResponse response = client.GetAcsResponse(request);
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRes
ponse.Content));
                // 过滤出文本反垃圾场景配置的文本库。
                List<DescribeKeywordLibResponse.DescribeKeywordLib_KeywordLib> allLibs =
response.KeywordLibList;
                List<DescribeKeywordLibResponse.DescribeKeywordLib_KeywordLib> textAntisp
amKeywordLibs = new List<DescribeKeywordLibResponse.DescribeKeywordLib_KeywordLib>();
                foreach (DescribeKeywordLibResponse.DescribeKeywordLib_KeywordLib keyword
Lib in allLibs)
                {
                    string LibType = keywordLib.LibType;
                    string resourceType = keywordLib.ResourceType;
                    string source = keywordLib.Source;
                    // 获取文本反垃圾自定义的关键词文本库
```

```
                    // 获取文本反垃圾自定义的关键词文本库。
                    if ("textKeyword".Equals(LibType) && "TEXT".Equals(resourceType) && "
MANUAL".Equals(source))
                    {
                        textAntispamKeywordLibs.Add(keywordLib);
                    }
                    // 获取文本反垃圾回流的关键词文本库。
                    if ("textKeyword".Equals(LibType) && "TEXT".Equals(resourceType) && "
FEEDBACK".Equals(source))
                    {
                        textAntispamKeywordLibs.Add(keywordLib);
                    }
                }
                Console.WriteLine(JsonConvert.SerializeObject(textAntispamKeywordLibs));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

- 查询相似文本库列表（包含自定义和系统回流的相似文本库）

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20170823;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            DescribeKeywordLibRequest request = new DescribeKeywordLibRequest();
            request.ServiceModule = "open_api";
            try
            {
                // 该方法将返回所有文本库，包括文本反垃圾的关键词文本库、文本反垃圾的相似文本文本库
、图片广告的关键词文本库、语音反垃圾的关键词文本库。
                DescribeKeywordLibResponse response = client.GetAcsResponse(request);
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRes
ponse.Content));
                // 过滤出相似文本文本库。
                List<DescribeKeywordLibResponse.DescribeKeywordLib_KeywordLib> allLibs =
response.KeywordLibList;
```

```
                List<DescribeKeywordLibResponse.DescribeKeywordLib_KeywordLib> similarTex
tLibs = new List<DescribeKeywordLibResponse.DescribeKeywordLib_KeywordLib>();
                foreach (DescribeKeywordLibResponse.DescribeKeywordLib_KeywordLib keyword
Lib in allLibs)
                {
                    String LibType = keywordLib.LibType;
                    String resourceType = keywordLib.ResourceType;
                    String source = keywordLib.Source;
                    // 获取文本反垃圾自定义的相似文本文本库。
                    if ("similarText".Equals(LibType) && "TEXT".Equals(resourceType) && "
MANUAL".Equals(source))
                    {
                        similarTextLibs.Add(keywordLib);
                    }
                    // 获取文本反垃圾回流的相似文本文本库。
                    if ("similarText".Equals(LibType) && "TEXT".Equals(resourceType) && "
FEEDBACK".Equals(source))
                    {
                        similarTextLibs.Add(keywordLib);
                    }
                }
                Console.WriteLine(JsonConvert.SerializeObject(similarTextLibs));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

创建文本库

- 创建关键词文本库

```
using System;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20170823;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            CreateKeywordLibRequest createKeywordLibRequest = new CreateKeywordLibRequest
();
            createKeywordLibRequest.ServiceModule = "open_api";
            createKeywordLibRequest.Name = "测试关键词文本库";
            // 设置为文本反垃圾场景使用。
            createKeywordLibRequest.ResourceType = "TEXT";
            // 设置类型为关键词。
            createKeywordLibRequest.LibType = "textKeyword";
            // 设置创建黑库。
            createKeywordLibRequest.Category = "BLACK";
            try
            {
                CreateKeywordLibResponse describeKeywordLibResponse = client.GetAcsRespon
se(createKeywordLibRequest);
                Console.WriteLine(System.Text.Encoding.Default.GetString(describeKeywordL
ibResponse.HttpResponse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

- 创建相似文本库

```
using System;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20170823;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            CreateKeywordLibRequest createKeywordLibRequest = new CreateKeywordLibRequest
();
            createKeywordLibRequest.ServiceModule = "open_api";
            createKeywordLibRequest.Name = "测试相似文本文本库";
            // 设置为文本反垃圾场景使用。
            createKeywordLibRequest.ResourceType = "TEXT";
            // 设置类型为相似文本。
            createKeywordLibRequest.LibType = "similarText";
            // 相似文本文本库支持创建黑名单库、白名单、疑似名单。
            createKeywordLibRequest.Category = "BLACK";
            try
            {
                CreateKeywordLibResponse describeKeywordLibResponse = client.GetAcsRespon
se(createKeywordLibRequest);
                Console.WriteLine(System.Text.Encoding.Default.GetString(describeKeywordL
ibResponse.HttpResponse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

修改文本库

更新文本库库名以及修改文本库所适用的业务场景（BizType）。

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20170823;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            UpdateKeywordLibRequest updateKeywordLibRequest = new UpdateKeywordLibRequest()
;
            // 设置要操作的文本库ID。
            updateKeywordLibRequest.Id = 文本库ID;
            // 设置新的文本库名称。
            updateKeywordLibRequest.Name = "测试修改名称";
            // 设置新的BizType。
            updateKeywordLibRequest.BizTypes = JsonConvert.SerializeObject(new List<string>
{ "comment", "title" });
            try
            {
                UpdateKeywordLibResponse updateKeywordLibResponse = client.GetAcsResponse(u
pdateKeywordLibRequest);
                Console.WriteLine(System.Text.Encoding.Default.GetString(updateKeywordLibRe
sponse.HttpResponse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

删除文本库

> **注意** 删除文本库也将删除文本库下的文本。系统回流的文本库不允许删除。

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20170823;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            DeleteKeywordLibRequest deleteKeywordLibRequest = new DeleteKeywordLibRequest()
;
            // 设置要删除的文本库ID。
            deleteKeywordLibRequest.Id = 文本库ID;
            try
            {
                DeleteKeywordLibResponse deleteKeywordLibResponse = client.GetAcsResponse(d
eleteKeywordLibRequest);
                Console.WriteLine(System.Text.Encoding.Default.GetString(deleteKeywordLibRe
sponse.HttpResponse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

查找文本

默认分页查询所有文本。如果设置了Keyword字段，将模糊查找包含该字段值的文本。

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20170823;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            DescribeKeywordRequest describeKeywordRequest = new DescribeKeywordRequest();
            // 要查询的文本库ID。
            describeKeywordRequest.KeywordLibId = 文本库ID;
            describeKeywordRequest.PageSize = 10;
            describeKeywordRequest.CurrentPage = 1;
            // 可选，用于模糊查。
            describeKeywordRequest.Keyword = "查询文本";
            try
            {
                DescribeKeywordResponse describeKeywordResponse = client.GetAcsResponse(des
cribeKeywordRequest);
                Console.WriteLine(System.Text.Encoding.Default.GetString(describeKeywordRes
ponse.HttpResponse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

添加文本

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20170823;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            CreateKeywordRequest createKeywordRequest = new CreateKeywordRequest();
            // 设置文本库ID。
            createKeywordRequest.KeywordLibId = 文本库ID;
            // 要添加的文本。
            createKeywordRequest.Keywords = JsonConvert.SerializeObject(new List<string> {
"文本_1", "文本_2" });
            try
            {
                CreateKeywordResponse createKeywordResponse = client.GetAcsResponse(createK
eywordRequest);
                Console.WriteLine(System.Text.Encoding.Default.GetString(createKeywordRespo
nse.HttpResponse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

删除文本

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20170823;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            DeleteKeywordRequest deleteKeywordRequest = new DeleteKeywordRequest();
            // 设置文本库ID。
            deleteKeywordRequest.KeywordLibId = "文本库ID";
            // 要删除的文本ID。
            deleteKeywordRequest.Ids = JsonConvert.SerializeObject(new List<int> { 文本ID_1,
文本ID_2 });
            try
            {
                DeleteKeywordResponse deleteKeywordResponse = client.GetAcsResponse(deleteK
eywordRequest);
                Console.WriteLine(System.Text.Encoding.Default.GetString(deleteKeywordRespo
nse.HttpResponse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

6.7.3. 自定义图库

本文介绍了如何使用.NET SDK管理自定义图库。

功能描述

您可以自定义智能鉴黄、暴恐涉政识别、图片或视频广告的图片样本，满足个性化内容管控需求。关于参数的详细信息，请参见创建图库API文档。

您需要使用内容安全的API接入地址，调用本SDK接口。关于API接入地址的信息，请参见接入地址（Endpoint）。

前提条件

已安装.NET依赖。关于安装.NET依赖的具体操作，请参见安装.NET依赖。

> **说明** 请一定按照安装.NET依赖页面中的版本安装，否则会导致调用失败。

查询自定义图库列表

您可以使用以下代码查询用户图库列表（包括用户自定义的图库列表和系统回流图库）：

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20170823;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            DescribeImageLibRequest describeImageLibRequest = new DescribeImageLibRequest()
;
            describeImageLibRequest.ServiceModule = "open_api";
            try
            {
                // 获取所有的图库，包括自定义的图库以及回流图库。
                DescribeImageLibResponse describeImageLibResponse = client.GetAcsResponse(d
escribeImageLibRequest);
                Console.WriteLine(System.Text.Encoding.Default.GetString(describeImageLibRe
sponse.HttpResponse.Content));
                List<DescribeImageLibResponse.DescribeImageLib_ImageLib> allLibs = describe
ImageLibResponse.ImageLibList;
                List<DescribeImageLibResponse.DescribeImageLib_ImageLib> customImageLibs =
new List<DescribeImageLibResponse.DescribeImageLib_ImageLib>();
                foreach (DescribeImageLibResponse.DescribeImageLib_ImageLib imageLib in all
Libs)
                {
                    String source = imageLib.Source;
                    // 过滤出用户自定义的图库。
                    if ("MANUAL".Equals(source))
                    {
                        customImageLibs.Add(imageLib);
                    }
                    // 过滤出系统回流（反馈）的图库。
                    if ("FEEDBACK".Equals(source))
                    {
                        customImageLibs.Add(imageLib);
                    }
```

```
                    Console.WriteLine(JsonConvert.SerializeObject(customImageLibs));
                }
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

创建自定义图库

您可以使用以下代码创建自定义图库：

> **说明** 请根据您所需要的业务场景设置不同的参数。

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20170823;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            CreateImageLibRequest createImageLibRequest = new CreateImageLibRequest();
            createImageLibRequest.ServiceModule = "open_api";
            createImageLibRequest.Name = "鉴黄黑库";
            createImageLibRequest.Scene = "PORN";
            createImageLibRequest.Category = "BLACK";
            try
            {
                CreateImageLibResponse createImageLibResponse = client.GetAcsResponse(creat
eImageLibRequest);
                Console.WriteLine(System.Text.Encoding.Default.GetString(createImageLibResp
onse.HttpResponse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

修改自定义图库

您可以使用以下代码修改自定义图库的名称及其适用的业务场景（BizType）：

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20170823;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            // IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            UpdateImageLibRequest updateImageLibRequest = new UpdateImageLibRequest();
            // 图库ID。
            updateImageLibRequest.Id = 图库ID;
            updateImageLibRequest.Name = "鉴黄黑库改名";
            updateImageLibRequest.BizTypes = JsonConvert.SerializeObject(new List<string> {
"comment" });
            updateImageLibRequest.Category = "WHITE";
            updateImageLibRequest.Scene = "PORN";
            try
            {
                UpdateImageLibResponse updateImageLibResponse = client.GetAcsResponse(updat
eImageLibRequest);
                Console.WriteLine(System.Text.Encoding.Default.GetString(updateImageLibResp
onse.HttpResponse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

删除自定义图库

您可以使用以下代码删除自定义图库：

> **注意** 删除自定义图库时，图库下的所有图片也将被删除。

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20170823;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            DeleteImageLibRequest deleteImageLibRequest = new DeleteImageLibRequest();
            // 图库ID。
            deleteImageLibRequest.Id = 图库ID;
            try
            {
                DeleteImageLibResponse deleteImageLibResponse = client.GetAcsResponse(delet
eImageLibRequest);
                Console.WriteLine(System.Text.Encoding.Default.GetString(deleteImageLibResp
onse.HttpResponse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

查询自定义图库图片列表

您可以使用以下代码查询自定义图库中所有已添加的图片列表：

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20170823;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            DescribeImageFromLibRequest describeImageFromLibRequest = new DescribeImageFrom
LibRequest();
            describeImageFromLibRequest.ImageLibId = 图库ID;
            describeImageFromLibRequest.PageSize = 20;
            describeImageFromLibRequest.CurrentPage = 1;
            try
            {
                DescribeImageFromLibResponse describeImageFromLibResponse = client.GetAcsRe
sponse(describeImageFromLibRequest);
                Console.WriteLine(System.Text.Encoding.Default.GetString(describeImageFromL
ibResponse.HttpResponse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

删除自定义图片

您可以使用以下代码删除自定义图库中的多张自定义图片：

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20170823;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            DeleteImageFromLibRequest deleteImageFromLibRequest = new DeleteImageFromLibReq
uest();
            deleteImageFromLibRequest.Ids = JsonConvert.SerializeObject(new List<int> { 图片
ID });
            try
            {
                DeleteImageFromLibResponse deleteImageFromLibResponse = client.GetAcsRespon
se(deleteImageFromLibRequest);
                Console.WriteLine(System.Text.Encoding.Default.GetString(deleteImageFromLib
Response.HttpResponse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

6.7.4. OSS内容检测

本文介绍了如何使用.NET SDK进行OSS违规检测。

前提条件

已安装.NET依赖。关于安装.NET依赖的具体操作，请参见安装.NET依赖。

> **说明** 请一定按照安装.NET依赖页面中的版本安装，否则会导致调用失败。

获取OSS违规检测数据

您可以参考以下代码示例获取OSS违规检测数据：

> **说明** 以下代码仅为简单示例，具体的接口参数请参考OSS违规检测API文档。

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20170823;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            DescribeOssResultItemsRequest describeKeywordLibRequest = new DescribeOssResult
ItemsRequest();
            describeKeywordLibRequest.ResourceType = "VIDEO";
            describeKeywordLibRequest.Scene = "porn";
            describeKeywordLibRequest.Stock = false;
            describeKeywordLibRequest.StartDate = "2021-10-11 00:00:00 +0800";
            describeKeywordLibRequest.EndDate = "2021-10-12 15:00:53 +0800";
            try
            {
                DescribeOssResultItemsResponse describeOssResultItemsResponse = client.GetA
csResponse(describeKeywordLibRequest);
                Console.WriteLine(System.Text.Encoding.Default.GetString(describeOssResultI
temsResponse.HttpResponse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

对OSS的审核结果进行标记

该接口能够对OSS的扫描结果进行标记和操作。如果需要对已检测出结果的内容执行删除、标记为正常并忽略，或者解除冻结等操作，您可以调用本接口。

> **说明** 以下代码仅为简单示例，具体的接口参数请参考OSS违规检测API。

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20170823;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            MarkOssResultRequest markOssResultRequest = new MarkOssResultRequest();
            markOssResultRequest.ResourceType = "VIDEO";
            markOssResultRequest.Scene = "terrorism";
            markOssResultRequest.Stock = false;
            markOssResultRequest.Ids = JsonConvert.SerializeObject(new List<long> { 2493000
1L });
            markOssResultRequest.Operation = "ignore";
            try
            {
                MarkOssResultResponse markOssResultResponse = client.GetAcsResponse(markOss
ResultRequest);
                Console.WriteLine(System.Text.Encoding.Default.GetString(markOssResultRespo
nse.HttpResponse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

以文件形式导出OSS违规检测结果

您可以参考以下代码示例通过文件形式导出OSS违规检测结果：

> **说明** 以下代码仅为简单示例，具体的接口参数请参考OSS违规检测API。

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20170823;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            ExportOssResultRequest exportOssResultRequest = new ExportOssResultRequest();
            exportOssResultRequest.ResourceType = "VIDEO";
            exportOssResultRequest.Scene = "porn";
            exportOssResultRequest.Stock = false;
            exportOssResultRequest.StartDate = "2021-10-11 00:00:00 +0800";
            exportOssResultRequest.EndDate = "2021-10-12 15:00:53 +0800";
            try
            {
                ExportOssResultResponse exportOssResultResponse = client.GetAcsResponse(exp
ortOssResultRequest);
                Console.WriteLine(System.Text.Encoding.Default.GetString(exportOssResultRes
ponse.HttpResponse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

6.7.5. 业务场景管理

6.7.5.1. 创建业务场景

本文介绍了如何通过.NET SDK创建业务场景，业务场景主要用于内容检测API功能自定义机审标准。

功能描述

业务场景主要用于内容检测API功能自定义机审标准，默认每个用户最多可以创建100个业务场景。关于参数的详细说明，请参见创建业务场景API文档。

前提条件

已安装.NET依赖。关于安装.NET依赖的具体操作，请参见安装.NET依赖。

> **说明** 请一定按照安装.NET依赖页面中的版本安装，否则会导致调用失败。

创建业务场景

接口	描述	支持的地域
CreateBizType	创建业务场景。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20170823;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            CreateBizTypeRequest createBizTypeRequest = new CreateBizTypeRequest();
            // 创建业务场景的名称。
            createBizTypeRequest.BizTypeName = "您要创建的业务场景";
            // 导入已经创建的业务场景对应的配置。可以不设置。
            createBizTypeRequest.BizTypeImport = "您已经创建的业务场景";
            // 是否引入行业模板配置。当值为true时必须传industryInfo。可以不设置。
            createBizTypeRequest.CiteTemplate = true;
            // 行业分类。CiteTemplate为true时，必须设置。取值范围：社交-注册信息-头像；社交-注册信息-昵称。
            createBizTypeRequest.IndustryInfo = "社交-注册信息-头像";
            // 指定API返回格式。
            createBizTypeRequest.AcceptFormat = FormatType.JSON;
            // 指定请求方法。
            createBizTypeRequest.Method = MethodType.POST;
            createBizTypeRequest.Encoding = "utf-8";
            // HTTP调用创建业务场景。
            try
```

```
            {
                CreateBizTypeResponse response = client.GetAcsResponse(createBizTypeRequest
);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

6.7.5.2. 删除业务场景

本文介绍了如何通过.NET SDK删除业务场景。如果您需要删除业务场景，请务必保证业务场景下无数据调用再删除，否则删除业务场景后对应的策略配置恢复为默认，有可能导致业务数据误识别。

功能描述

关于参数的详细说明，请参见删除业务场景API文档。

前提条件

已安装.NET依赖。关于安装.NET依赖的具体操作，请参见安装.NET依赖。

> **说明** 请一定按照安装.NET依赖页面中的版本安装，否则会导致调用失败。

删除业务场景

接口	描述	支持的地域
DeleteBizType	删除业务场景。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20170823;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            DeleteBizTypeRequest deleteBizTypeRequest = new DeleteBizTypeRequest();
            // 设置需要删除业务场景名称。
            deleteBizTypeRequest.BizTypeName = "您要删除的业务场景";
            // 指定API返回格式。
            deleteBizTypeRequest.AcceptFormat = FormatType.JSON;
            // 指定请求方法。
            deleteBizTypeRequest.Method = MethodType.POST;
            deleteBizTypeRequest.Encoding = "utf-8";
            try
            {
                DeleteBizTypeResponse response = client.GetAcsResponse(deleteBizTypeRequest
);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

6.7.5.3. 查询业务场景

本文介绍了如何通过.NET SDK查询已创建和自定义的业务场景列表，用于在后台管理业务场景数据。

功能描述

关于参数的详细说明，请参见查询业务场景API文档。

前提条件

已安装.NET依赖。关于安装.NET依赖的具体操作，请参见安装.NET依赖。

> **说明** 请一定按照安装.NET依赖页面中的版本安装，否则会导致调用失败。

查询业务场景

接口	描述	支持的地域
DescribeUserBizTypes	查询业务场景。	• cn-shanghai：华东2（上海） • cn-beijing：华北2（北京） • cn-shenzhen：华南1（深圳） • ap-southeast-1：新加坡

示例代码

```
using System;
using Newtonsoft.Json;
using Aliyun.Acs.Core;
using Aliyun.Acs.Core.Http;
using Aliyun.Acs.Core.Profile;
using Aliyun.Acs.Green.Model.V20170823;
using System.Collections.Generic;
namespace csharp_sdk_sample
{
    class Program
    {
        static void Main(string[] args)
        {
            // 请替换成您的AccessKey ID、AccessKey Secret。
            IClientProfile profile = DefaultProfile.GetProfile(
                "cn-shanghai",
                "您的AccessKey ID",
                "您的AccessKey Secret");
            DefaultAcsClient client = new DefaultAcsClient(profile);
            DescribeUserBizTypesRequest describeUserBizTypesRequest = new DescribeUserBizTy
pesRequest();
            // 指定API返回格式。
            describeUserBizTypesRequest.AcceptFormat = FormatType.JSON;
            // 指定请求方法。
            describeUserBizTypesRequest.Method = MethodType.GET;
            describeUserBizTypesRequest.Encoding = "utf-8";
            try
            {
                DescribeUserBizTypesResponse response = client.GetAcsResponse(describeUserB
izTypesRequest);
                if (response.HttpResponse.Status != 200)
                {
                    Console.WriteLine("the request failed. status:{0}", response.HttpRespon
```

```
se.Status);
                }
                Console.WriteLine(System.Text.Encoding.Default.GetString(response.HttpRespo
nse.Content));
                // 所有业务场景列表。
                Console.WriteLine("query success. bizTypes size :" + response.BizTypeList.C
ount);
                foreach (DescribeUserBizTypesResponse.DescribeUserBizTypes_Item bizTypeItem
in response.BizTypeList)
                {
                    // 业务场景名称。
                    Console.WriteLine(bizTypeItem.BizType);
                    // 是否属于引用行业模板。
                    Console.WriteLine(bizTypeItem.CiteTemplate);
                    // 行业信息。
                    Console.WriteLine(bizTypeItem.IndustryInfo);
                    // 业务场景来源。客户自定义：custom；内容安全服务默认配置：system。
                    Console.WriteLine(bizTypeItem.Source);
                }
            }
            catch (Exception ex)
            {
                Console.WriteLine("Failed with error info: {0}", ex.Message);
            }
        }
    }
}
```

7.其他语言SDK

如果我们提供的Java、PHP(5.3+)、Python(2.7+) SDK无法满足您对开发语言的要求，建议您根据调用方式和具体接口文档构建HTTP请求，调用我们的服务。更多内容，请参见调用方式和API文档。

由于开发语言种类繁多，我们无法熟悉和精通各种开发语言并维护相应的SDK，但是我们收集了第三方开发者编写的内容安全SDK供您参考。内容安全第三方SDK包含以下开发语言：C#、C++、Nodejs、Python(3.5)、Go。

> **说明** 第三方SDK仅作列举参考，阿里云暂时不提供后续维护。如有问题，请您根据开发准备修改。更多内容，请参见内容安全开发准备。

单击下载内容安全第三方SDK（c#、c++、nodejs、python(3.5)、go）

8.代码示例下载

参见SDK 概览中的开发准备部分，下载JAVA、PHP、Python调用示例，以作参考。

9.SDK更新历史

本文介绍了内容安全SDK的更新历史记录。

最新SDK版本：aliyun-java-sdk-green 3.6.5

最近更新时间：2021-04-14

上一个SDK版本：aliyun-java-sdk-green 3.6.4

单击查看历史版本文档。

发布记录

发布时间	更新对象	版本	更新说明
2021-04-14	aliyun-java-sdk-green	3.6.5	历史稳定性更新。
2020-12-04	aliyun-java-sdk-green	3.6.3	新增手动检测接口。
2020-11-18	aliyun-java-sdk-green	3.6.2	新增网页检测能力。
2020-08-03	aliyun-java-sdk-green	3.6.1	• 修复一些接口返回结构体问题。 • 稳定性更新。
2020-03-23	aliyun-java-sdk-green	3.5.2	新增获取账单接口。
2019-10-09	aliyun-java-sdk-green	3.5.1	新增取消视频检测接口。
2019-08-19	aliyun-java-sdk-green	3.5.0	新增人脸属性检测接口。
2018-11-21	aliyun-java-sdk-green	3.4.1	• 支持语音流、语音文件检测。 • 支持本地文件上传检测。 • 支持视频直播流检测。
2018-09-04	aliyun-java-sdk-green	3.3.0	新增声纹识别接口。
2017-05-12	aliyun-java-sdk-green	3.0.0	全新设计的sdk，更加易用和标准化。 • 统一输入参数和输出参数。 • 更新图片鉴黄为分类检测接口。 • 支持GIF，并对长图检测做优化处理。

发布时间	更新对象	版本	更新说明
2016-12-16	aliyun-java-sdk-green	2.6.0	更新内容： • 文本垃圾检测返回风险分类。 • 支持指定关键词ID。 • 图片检测支持牛皮癣广告识别（公测）。 • 修复人脸接口返回人脸位置解析问题。
2016-10-18	aliyun-java-sdk-green	2.1.0	版本更新至2.1.0。
2016-06-21	aliyun-java-sdk-green	1.4.0	版本更新至1.4.0。
2016-03-08	aliyun-java-sdk-green	1.1.0	版本更新至1.1.0。
2016-02-24	aliyun-java-sdk-green	1.0.3	版本更新至1.0.3。
2016-01-18	aliyun-java-sdk-green	1.0.2	版本更新至1.0.2。
2015-12-22	aliyun-java-sdk-green	1.0.1	版本更新至1.0.1。
2015-12-16	aliyun-java-sdk-green	1.0.0	版本1.0.0正式上线。