ALIBABA CLOUD

# Alibaba Cloud

## 内容安全

## API Reference

Document Version: 20220622

Alibaba Cloud

# Legal disclaimer

Alibaba Cloud reminds you to carefully read and fully understand the terms and conditions of this legal disclaimer before you read or use this document. If you have read or used this document, it shall be deemed as your total acceptance of this legal disclaimer.

1. You shall download and obtain this document from the Alibaba Cloud website or other Alibaba Cloud-authorized channels, and use this document for your own legal business activities only. The content of this document is considered confidential information of Alibaba Cloud. You shall strictly abide by the confidentiality obligations. No part of this document shall be disclosed or provided to any third party for use without the prior written consent of Alibaba Cloud.

2. No part of this document shall be excerpted, translated, reproduced, transmitted, or disseminated by any organization, company or individual in any form or by any means without the prior written consent of Alibaba Cloud.

3. The content of this document may be changed because of product version upgrade, adjustment, or other reasons. Alibaba Cloud reserves the right to modify the content of this document without notice and an updated version of this document will be released through Alibaba Cloud-authorized channels from time to time. You should pay attention to the version changes of this document as they occur and download and obtain the most up-to-date version of this document from Alibaba Cloud-authorized channels.

4. This document serves only as a reference guide for your use of Alibaba Cloud products and services. Alibaba Cloud provides this document based on the "status quo", "being defective", and "existing functions" of its products and services. Alibaba Cloud makes every effort to provide relevant operational guidance based on existing technologies. However, Alibaba Cloud hereby makes a clear statement that it in no way guarantees the accuracy, integrity, applicability, and reliability of the content of this document, either explicitly or implicitly. Alibaba Cloud shall not take legal responsibility for any errors or lost profits incurred by any organization, company, or individual arising from download, use, or trust in this document. Alibaba Cloud shall not, under any circumstances, take responsibility for any indirect, consequential, punitive, contingent, special, or punitive damages, including lost profits arising from the use or trust in this document (even if Alibaba Cloud has been notified of the possibility of such a loss).

5. By law, all the contents in Alibaba Cloud documents, including but not limited to pictures, architecture design, page layout, and text description, are intellectual property of Alibaba Cloud and/or its affiliates. This intellectual property includes, but is not limited to, trademark rights, patent rights, copyrights, and trade secrets. No part of this document shall be used, modified, reproduced, publicly transmitted, changed, disseminated, distributed, or published without the prior written consent of Alibaba Cloud and/or its affiliates. The names owned by Alibaba Cloud shall not be used, published, or reproduced for marketing, advertising, promotion, or other purposes without the prior written consent of Alibaba Cloud. The names owned by Alibaba Cloud include, but are not limited to, "Alibaba Cloud", "Aliyun", "HiChina", and other brands of Alibaba Cloud and/or its affiliates, which appear separately or in combination, as well as the auxiliary signs and patterns of the preceding brands, or anything similar to the company names, trade names, trademarks, product or service names, domain names, patterns, logos, marks, signs, or special descriptions that third parties identify as Alibaba Cloud and/or its affiliates.

6. Please directly contact Alibaba Cloud for any errors of this document.

# Document conventions

| Style | Description | Example |
|---|---|---|
| Danger | A danger notice indicates a situation that will cause major system changes, faults, physical injuries, and other adverse results. | Danger:<br>Resetting will result in the loss of user configuration data. |
| Warning | A warning notice indicates a situation that may cause major system changes, faults, physical injuries, and other adverse results. | Warning:<br>Restarting will cause business interruption. About 10 minutes are required to restart an instance. |
| Notice | A caution notice indicates warning information, supplementary instructions, and other content that the user must understand. | Notice:<br>If the weight is set to 0, the server no longer receives new requests. |
| Note | A note indicates supplemental instructions, best practices, tips, and other content. | Note:<br>You can use Ctrl + A to select all files. |
| > | Closing angle brackets are used to indicate a multi-level menu cascade. | Click **Settings**> **Network**> **Set network type**. |
| **Bold** | Bold formatting is used for buttons , menus, page names, and other UI elements. | Click **OK**. |
| `Courier font` | Courier font is used for commands | Run the `cd /d C:/window` command to enter the Windows system folder. |
| *Italic* | Italic formatting is used for parameters and variables. | `bae log list --instanceid`<br>*Instance_ID* |
| [] or [a\|b] | This format is used for an optional value, where only one item can be selected. | `ipconfig [-all\|-t]` |
| {} or {a\|b} | This format is used for a required value, where only one item can be selected. | `switch {active\|stand}` |

# Table of Contents

# 1.Development preparations

Before you activate Alibaba Cloud Content Moderation, you must familiarize yourself with the service agreement, billing methods, and limits, and complete development preparations.

## Get familiar with the service agreement and billing methods

Make sure that you have fully understood the service agreement and billing methods before you use the Content Moderation API.

## Get started with the Content Moderation API

Before you use the Content Moderation API, you must register an Alibaba Cloud account, create an AccessKey pair, and activate Alibaba Cloud Content Moderation.

Procedure

1. Go to the Alibaba Cloud official website and register an account. Skip this step if you have registered an account.
2. Go to the landing page of Alibaba Cloud Content Moderation and click **Enable Now** to activate the service.
3. Manage your AccessKey ID and AccessKey secret, For more information, see Obtain an AccessKey pair.

> ? **Note** An AccessKey pair is the key for using the Content Moderation API and has all permissions on your Alibaba Cloud account. Keep your AccessKey pair confidential and change it periodically.

## Get familiar with usage limits

Before using the content detection API, it is recommended that you understand the concurrency limit. For more information about concurrency limits, see FAQ about the Content Moderation API

# 2.List of operations by function

The following tables list the API operations available for use in Alibaba Cloud Content Moderation.

## Make API requests

This topic applies to users who initiate HTTP or HTTPS POST requests by using request URLs. If you use an SDK to initiate HTTP or HTTPS POST requests, you can skip this topic.

Each request URL consists of different parameters and has a fixed syntax. Generally, a URL contains common parameters, your signature, and operation-specific parameters. Sample request URLs are provided for each operation. These URLs are not encoded to make them easy to read. Before you make a request, you must encode the request URL. A response is returned to you after the authentication is complete based on your signature. Response parameters are returned if the call is successful, whereas an error message is returned if the call fails. You can troubleshoot issues based on the common error codes and operation-specific error codes.

> ⓘ **Note** We recommend that you use an SDK to call an operation and manage your resources. If you use an SDK, manual signature verification is not required.

## Image moderation

| Operation | Description |
|---|---|
| /green/image/scan | Submits image moderation tasks to moderate images for illegal or undesirable content and obtains results in real time. |
| /green/image/asyncscan | Submits asynchronous image moderation tasks to moderate images for illegal or undesirable content. |
| /green/image/results | Queries asynchronous image moderation results. |
| /green/image/feedback | Gives feedback on image moderation results. |

## Image OCR

| Operation | Description |
|---|---|
| /green/image/scan | Submits optical character recognition (OCR) tasks to detect text in images and obtains results in real time. |
| /green/image/asyncscan | Submits asynchronous OCR tasks to detect text in images. |
| /green/image/results | Queries asynchronous OCR results. |

## Video moderation

| Operation | Description |
|---|---|
| /green/video/syncscan | Submits video moderation tasks to detect illegal or undesirable content in videos and obtains results in real time. |
| /green/video/asyncscan | Submits asynchronous video moderation tasks to detect illegal or undesirable content in videos. |
| /green/video/results | Queries asynchronous video moderation results. |
| /green/video/feedback | Gives feedback on video moderation results. |

## Text anti-spam

| Operation | Description |
|---|---|
| /green/text/scan | Submits text moderation tasks to detect illegal content in text. |
| /green/text/feedback | Gives feedback on text moderation results. |

# 3.Endpoint

This topic describes the endpoints of content moderation services.

The endpoint of a content moderation service is in the following format:

```
http(s)://green.{region}.aliyuncs.com/Operation
```

Where,

- `{region}` indicates a supported region. The following table lists the supported regions.

| Region | Code | Public network endpoint | Support HTTPS or not | Support internal network access from ECS or not |
|---|---|---|---|---|
| China (Shanghai) | cn-shanghai | green.cn-shanghai.aliyuncs.com | Supported only by SDK V3.0.0 or later | Not supported |
| China (Beijing) | cn-beijing | green.cn-beijing.aliyuncs.com | Supported only by SDK V3.0.0 or later | Not supported |
| China (Shenzhen) | cn-shenzhen | green.cn-shenzhen.aliyuncs.com | Supported only by SDK V3.0.0 or later | Not supported |
| Singapore | ap-southeast-1 | green.ap-southeast-1.aliyuncs.com | Supported only by SDK V3.0.0 or later | Not supported |
| US (Silicon Valley) | us-west-1 | green.us-west-1.aliyuncs.com | Supported only by SDK V3.0.0 or later | Not supported |
| Indonesia | ap-southeast-5 | green.ap-south-east-5 | Supported only by SDK V3.0.0 or later | Not supported |

- `Operation` indicates the URI of the specific operation. For more information, see the description of each operation.

For example, the following endpoint points to the pornography detection service in the China (Shanghai) region:

```
http(s)://green.cn-shanghai.aliyuncs.com/green/image/scan
```

# 4.Getting started

## 4.1. Request structure

This topic describes the structure of the HTTP and HTTPS POST requests sent to call the Content Moderation API.

### Methods

The Content Moderation API supports both HTTP and HTTPS POST requests. The JSON-formatted request parameters are sent to the server in the form of an HTTP request body. The common query parameter clientInfo is available in the URL query parameters.

The following is an example of an unencoded synchronous request for moderating images:

```
http://green.cn-shanghai.aliyuncs.com/green/image/scan?clientInfo={"userId":"120234234","us
erNick":"Mike","userType":"others"}
```

Where,

- `http` specifies the protocol for transmitting the request.
- `green.cn-shanghai.aliyuncs.com` specifies the endpoint of the Content Moderation API.
- `green/image/scan` specifies a specific operation, that is, the operation for synchronous image moderation.
- `clientInfo={"ip":"127.0.0.2","userId":"120234234","userNick":"Mike","userType":"others"}` specifies the common query parameters.

### Protocols

You can send requests over HTTP or HTTPS. With regard to higher security, we recommend that you send requests over HTTPS, especially when sensitive data is involved.

### Parameters

In each request, you must specify common request parameters and operation-specific service request parameters. Service parameters are transparently transmitted in the request body. For more information about the body structure, see the description of each operation.

### Encoding

All requests and responses are encoded in UTF-8.

## 4.2. Common parameters

Common parameters are parameters that are included in all HTTP requests. Common parameters consist of common request headers and common request parameters.

### Common request headers

The following table describes common request headers that must be used when you send HTTP POST requests by using URLs to call the Content Moderation API.

| Header | Type | Required | Description |
|---|---|---|---|
| Accept | String | Yes | The data type of the response that can be accepted. Only the JSON type is supported. Set the value to **application/json**. |
| Content-Type | String | Yes | The data type of the request body. Only the JSON type is supported. Set the value to **application/json**. |
| Content-MD5 | String | No | The encoded value that is computed based on the request body.<br>To obtain the value, use the content of the request body as a string, apply the MD5 algorithm on the string to obtain an output of binary data, and then encode the output based on the Base64 algorithm.<br>**Note** This header is optional when the **HMAC-SM3** encryption algorithm is used. However, this header is required when the **HMAC-SHA1** encryption algorithm is used. By default, the HMAC-SHA1 encryption algorithm is used. |
| Date | String | Yes | The time when the request is sent. This header uses the GMT format. Example: `Tue, 17 Jan 2017 10:16:36 GMT`. |
| x-acs-version | String | Yes | The version of the Content Moderation API. Set the value to **2018-05-09**. |
| x-acs-signature-nonce | String | Yes | A unique, random number that is used to prevent replay attacks. We recommend that you use different numbers for different requests. |
| x-acs-signature-version | String | Yes | The version of the signature encryption algorithm. Set the value to **1.0**. |
| x-acs-signature-method | String | Yes | The encryption method of the signature string. Valid values:<br>- **HMAC-SHA1**: This is the default value.<br>- **HMAC-SM3**<br>For more information about the encryption method, see Signature method. |
| x-acs-content-sm3 | String | No | The encoded value that is computed based on the request body.<br>To obtain the value, use the content of the request body as a string, apply the HMAC-SM3 algorithm on the string to obtain a byte array, and then convert the array into the hexadecimal format. This is the encodeHexString(SM3Digest(body)) method. |

| Header | Type | Required | Description |
|---|---|---|---|
| Authorization | String | Yes | The authentication method. The format is `"acs" + " " + AccessKeyId + ":" + signature`.<br>Parameter description:<br>- `AccessKeyId`: the AccessKey ID that you can obtain from the Alibaba Cloud Management Console. For more information about how to obtain the AccessKey ID, see Obtain an AccessKey pair.<br>- `signature`: the request signature. For more information about the signature encryption algorithm, see Signature method. |

## Common request parameters

The following table describes common request parameters that can be used when you send HTTP POST requests by using URLs to call the Content Moderation API.

| Parameter | Type | Required | Description |
|---|---|---|---|
| clientInfo | String | No | The information about the client that sends the request. The information is serialized from the ClientInfo structure in the JSON format. The information includes the unique machine identifier (UMID) and International Mobile Equipment Identity (IMEI) of the client. For more information, see ClientInfo. |

ClientInfo

| Parameter | Type | Required | Description |
|---|---|---|---|
| sdkVersion | String | No | The version of the Content Moderation SDK.<br>This parameter is required when an operation is called by using the Content Moderation SDK. For more information, see SDK overview. |
| cfgVersion | String | No | The version of the configuration.<br>This parameter is required when an operation is called by using the Content Moderation SDK. For more information, see SDK overview. |
| userType | String | No | The type of the user account. Valid values:<br>- **taobao**: indicates a Taobao account.<br>- **others**: indicates an account other than a Taobao account. |

| Parameter | Type | Required | Description |
|---|---|---|---|
| userId | String | No | The unique ID of the user in the business system.<br>We strongly recommend that you set this parameter to analyze the violations of the user. |
| userNick | String | No | The nickname of the user. |
| avatar | String | No | The URL that can be used to access the profile picture of the user over the Internet. You can pass the value of this parameter to analyze the behaviors of the user. |
| imei | String | No | The IMEI of the device. |
| imsi | String | No | The International Mobile Subscriber Identity (IMSI) of the device. |
| umid | String | No | The UMID of the device. |
| ip | String | No | The public IP address of the user in the business system.<br>If the request is initiated from the device, this parameter can be left empty. If the request is initiated from the server, this parameter is set to the logon IP address of the user or the public IP address of the device.<br>We recommend that you set this parameter to analyze the violations of the user. If this parameter is not specified in the request, the server attempts to obtain the IP address from the request URL or HTTP request headers. |
| os | String | No | The operating system of the device. Example: *Android 6.0*. |
| channel | String | No | The channel number. |
| hostAppName | String | No | The name of the host application. |
| hostPackage | String | No | The name of the host application package. |
| hostVersion | String | No | The version of the host application. |

## Sample requests

```
POST /green/image/scan?clientInfo=%7b%22userId%22%3a%22120234234%22%2c%22userNick%22%3a%22M
ike%22%2c%22userType%22%3a%22others%22%7d HTTP/1.1
Accept: application/json
Content-Type: application/json
Content-MD5: C+5Y0crpO4sYgC2DNjycug==
Date: Tue, 15 May 2018 10:16:36 GMT
x-acs-version: 2018-05-09
x-acs-signature-nonce: adfkjfaHKslkjf
x-acs-signature-version: 1.0
x-acs-signature-method: HMAC-SHA1
Authorization: acs q23Q9K4PaGi12Wyk:adFff0FFO6sGcDbDNjRtso==
Host: green.cn-shanghai.aliyuncs.com/
Content-Length: 191
{
    "scenes": [
        "porn"
    ],
    "tasks": [
        {
            "dataId": "test2NInmO$tAON6qYUrtCRgLo-1mwxdi",
            "url": "https://img.alicdn.com/tfs/TB1urBOQFXXXXbMXFXXXXXXXXXX-1442-257.png"
        }
    ]
}
```

Parameter description:

- `POST` specifies the request method.
- `/green/image/scan` is the request path.
- `clientInfo=%7b%22userId%22%3a%22120234234%22%2c%22userNick%22%3a%22Mike%22%2c%22userType%22%3a%22others%22%7d` is the encoded common request parameters.
- The part from `Accept:application/json` to `Content-Length:191` are HTTP request headers. In this part, the headers from `Accept: application/json` to `Authorization: acs q23Q9K4PaGi12Wyk:adFff0FFO6sGcDbDNjRtso==` are the common request headers used for signature.
- The content enclosed in braces `{}` is the JSON-formatted request body. The structure of the request body depends on the definition of different API operations. For more information, see List of operations by function.

# 4.3. Signature method

This topic describes the signature method that you can use to call the Content Moderation API.

## Procedure

1. Serialize request headers.

   Concatenate all the request headers that start with `x-acs-` to create a string based on the following rules:

   i. Extract all the HTTP headers that start with `x-acs-`.

   ii. Sort the extracted HTTP headers in alphabetical order.

iii. Write each HTTP header in the format of `"HTTP header name" + ":" + "HTTP header value" + "\n"`.

2. Serialize the URI and query parameters.

   Use the `uri + "?clientInfo =" + "JSON string of ClientInfo"` format to concatenate the URI and `clientInfo` parameters.

   > Note In this step, URL encoding is not required.

3. Create a string-to-sign.

   Use the following syntax:

```
"POST\napplication/json\n" +
"Value of the Content-MD5 header" + "\n" +
"application/json" + "\n" +
"Value of the Date header" + "\n" +
"Serialized request headers" +
"Serialized URI and query parameters"
```

   The following example shows a complete string-to-sign that can be used to call a synchronous operation for image moderation:

```
POST
application/json
C+5Y0crpO4sYgC2DNjycug==
application/json
Tue, 14 Mar 2017 06:29:50 GMT
x-acs-signature-method:HMAC-SHA1
x-acs-signature-nonce:339497c2-d91f-4c17-a0a3-1192ee9e2202
x-acs-signature-version:1.0
x-acs-version:2018-05-09
/green/image/scan? clientInfo={"ip":"127.xxx.xxx.2","userId":"12023xxxx","userNick":"Mi
ke","userType":"others"}
```

4. Generate the signature string.

   i. Use the HMAC-SHA1 algorithm to calculate the HMAC value of the string-to-sign that is created in Step 3. The AccessKey secret is used as the key for the HMAC calculation. The HMAC value is an array of bytes.

   > Note In this step, do not perform other operations on the HMAC value.

   ii. Encode the HMAC value in Base64 to obtain the signature string.

   iii. Set the value of `signature` to the obtained signature string in the `Authorization` header. The value of the Authorization header follows the syntax: `"acs" + " " + AccessKeyId + ":" + signature`.

   > Note In the preceding syntax, a space is added between `acs` and `AccessKeyId`.

# 4.4. Common response parameters

This topic describes the common response parameters and HTTP status codes for calling content moderation operations.

## Common response parameters

The following table describes the common parameters in the JSON data returned upon receiving any request.

| Parameter | Type | Required | Description |
|---|---|---|---|
| code | Integer | Yes | The HTTP status code returned for the moderation task. Some extended error codes are supported.<br>● 2xx indicates a success.<br>● 4xx indicates an incorrect request.<br>● 5xx indicates a back-end error.<br>For more information, see Common HTTP status codes. |
| msg | String | Yes | The message returned for the moderation task. |
| requestId | String | Yes | The unique ID of the request, which can be used to locate issues. |
| data | Integer, String, or JSON object | No | The service-related response data. This parameter may be empty when an error occurs. Generally, the value is a JSON structure or array. |

## Common HTTP status codes

| Identifier | HTTP status code | Description |
|---|---|---|
| OK | 200 | The message returned because the request is successful. |
| PROCESSING | 280 | The error message returned because the task is running. We recommend that you wait for a while, for example, 5 seconds, before querying the result. |
| BAD_REQUEST | 400 | The error message returned because the request is incorrect. This error is typically caused by incorrect request parameters. Check the request parameters carefully. |
| NOT_ALLOWED | 401 | The error message returned because an insecure image or video link is used. |
| FORBIDDEN | 403 | The error message returned because your image or video link cannot be accessed. Check whether the link is accessible from the Internet and whether an anti-theft chain policy is configured. |

| Identifier | HTTP status code | Description |
|---|---|---|
| NOT_FOUND | 404 | The error message returned because the content to be moderated cannot be found. This is usually because your image or video content cannot be downloaded. Make sure that the content is accessible from the Internet. |
| DOWNLOAD_FAILED | 480 | The error message returned because the download fails. Make sure that the size and resolution, if any, of the content to be moderated are within the limits of the API. |
| GENERAL_ERROR | 500 | The error message returned because a temporary server error occurs. We recommend that you try again. If the error code persists, submit a ticket. |
| DB_FAILED | 580 | The error message returned because an error occurs while operating on the database. We recommend that you try again. If the error code persists, submit a ticket. |
| TIMEOUT | 581 | The error message returned because the operation times out. We recommend that you try again. If the error code persists, submit a ticket. |
| CACHE_FAILED | 585 | The error message returned because a cache error occurs. We recommend that you try again. If the error code persists, submit a ticket. |
| ALGO_FAILED | 586 | The error message returned because an algorithm error occurs. Submit a ticket to us. |
| MQ_FAILED | 587 | The error message returned because a middleware error occurs. Submit a ticket to us. |
| EXCEED_QUOTA | 588 | The error message returned because the number of requests exceeds the quota. The quota defaults to 50 images, 20 videos, 20 audios, or 100 pieces of text per second. If you want to adjust the quota, submit a ticket. |
| TOO_LARGE | 589 | The error message returned because the content to be moderated is too large in size. Make sure that the content size is allowed by the API. We recommend that you try again. If the error code persists, submit a ticket. |
| BAD_FORMAT | 590 | The error message returned because the content to be moderated is in an incorrect format. Make sure that the format of the content is allowed by the API. |
| CONNECTION_POOL_FULL | 591 | The error message returned because the connection pool is full. Submit a ticket to us. |
| DOWNLOAD_TIMEOUT | 592 | The error message returned because the download times out. The download duration is limited to 3 seconds. Make sure that the size of the moderated content is allowed by the API. |

| Identifier | HTTP status code | Description |
|---|---|---|
| EXPIRED | 594 | The error message returned because the task expires. For example, taskId expires. |
| CATCH_FRAME_FAILED | 595 | The error message returned because frame capturing fails. Submit a ticket to us. |
| PERMISSION_DENY | 596 | The error message returned because the account is unauthorized, overdue, inactivated, or disabled. For more information, see the returned message. |

# 5.Content Moderation (machine moderation)

## 5.1. Image moderation

### 5.1.1. Synchronous moderation

This topic describes the /green/image/scan operation that you can call to moderate images and obtain moderation results in real time. You can call this operation to moderate images for risky or illegal content. This operation is applicable to the following scenarios: pornography detection, terrorist content detection, ad violation detection, QR code detection, undesirable scene detection, and logo detection.

#### Operation description

Operation: /green/image/scan

You can call this operation to submit image moderation tasks and obtain moderation results in real time.

- **Billing method:**
- **Response timeout:**
- **Return results:**
- **Limits on images:**

Scenarios

| Scenario | Description | Category of the moderation result |
|---|---|---|
| Pornography detection | Detects pornographic or sexy content in images. | normal, sexy, and porn |
| Terrorist content detection | Detects terrorist or political content in images. | normal, bloody, explosion, outfit, logo, weapon, politics, violence, crowd, parade, carcrash, flag, location, and others |
| Ad violation detection | Detects ads or illegal text in images. | normal, ad, politics, porn, abuse, terrorism, contraband, spam, npx, qrcode, and programCode<br>**Note** You can customize a policy for machine-assisted moderation and specify the categories that you want to detect based on actual requirements. For more information, see Customize policies for machine-assisted moderation. |

| Scenario | Description | Category of the moderation result |
|---|---|---|
| QR code detection | Detects QR codes or mini program codes in images. | normal, qrcode, and programCode<br>Note You can customize a policy for machine-assisted moderation and specify the categories that you want to detect based on actual requirements. For more information, see Customize policies for machine-assisted moderation. |
| Undesirable scene detection | Detects undesirable scenes, such as black screen, black edge, dark image, picture-in-picture, smoking, and live broadcasting in a running vehicle, in images. | normal, meaningless, PIP, smoking, and drivelive |
| Logo detection | Detects logos, such as TV station logos and trademarks, in images. | normal, TV, and trademark |

## Request parameters

| Parameter | Type | Required | Example | Description |
|---|---|---|---|---|
| bizType | String | No | default | |
| scenes | String Array | Yes | ["porn","terrorism","ad","live","qrcode","logo"] | The moderation scenario. Valid values:<br>- *porn*: pornography detection<br>- *terrorism*: terrorist content detection<br>- *ad*: ad violation detection<br>- *qrcode*: QR code detection<br>- *live*: undesirable scene detection<br>- *logo*: logo detection<br>You can specify multiple scenarios. For example, you can specify both porn and terrorism in the `scenes` parameter to detect pornographic and terrorist content.<br>Note If your moderation occurs in multiple scenarios at a time, you are charged the cumulative fee of all scenarios. The fee of each scenario equals the number of images that are moderated in the scenario multiplied by the unit price of the scenario. |

| Parameter | Type | Required | Example | Description |
|---|---|---|---|---|
| tasks | JSONArray | Yes | | The list of moderation objects. Each element in the JSON array is a moderation task structure. The JSON array can contain a maximum of 100 elements. In other words, you can submit a maximum of 100 images at a time. To submit 100 images at a time, you must raise the relevant concurrency limit to a number greater than 100. For more information about the structure of each element, see task. |

task

| Parameter | Type | Required | Example | Description |
|---|---|---|---|---|
| clientInfo | JSONObject | No | null | |
| dataId | String | No | uuid-xxxx-xxxx-1234 | |
| url | String | Yes | http://example.com/xxx.jpg | The URL of the image to be moderated. |
| extras | JSONObject | No | {"hitLibInfo":[{"context":"Haokan","libCode":"2144002","libName":"Text pattern library a"}]} | The extra parameters that you can set for calling the operation. This parameter is not required for image moderation. |

| Parameter | Type | Required | Example | Description |
|---|---|---|---|---|
| interval | Integer | No | 2 | The interval between two frames that are consecutively captured. This parameter is dedicated for GIF or long image moderation.<br>• A GIF image can be regarded as an array of frames. One frame is captured for moderation from every n frames. The n is specified by the interval parameter. The system captures frames from GIF images only when this parameter is specified.<br>• Long images can be in portrait or horizontal mode.<br>○ To moderate a long portrait image, in which the height is greater than 400 pixels and the ratio of height to width is greater than 2.5, you can divide the height by the width and round up the result to the nearest integer as the total number of frames.<br>○ To moderate a long horizontal image, in which the width is greater than 400 pixels and the ratio of width to height is greater than 2.5, you can divide the width by the height and round up the result to the nearest integer as the total number of frames.<br>By default, only the first frame of a GIF image or a long image is moderated. You can use the interval parameter to specify the interval between two frames that the system consecutively captures. This helps reduce moderation costs.<br>**Note** The interval and maxFrames parameters must be used in pairs. For example, the interval parameter is set to 2 and the maxFrames parameter is set to 10 for moderating a GIF image or a long image. In this example, one out of every two frames is moderated and a maximum of 10 frames are moderated. The fee is calculated based on the actual number of moderated frames. |
| maxFrames | Integer | No | 10 | |

## Response parameters

| Parameter | Type | Example | Description |
|---|---|---|---|
| code | Integer | 200 | |
| msg | String | OK | The message that is returned for the request. |
| dataId | String | uuid-xxxx-xxxx-1234 | |
| taskId | String | img4wlJcb7p4wH4lAP3111111-123456 | The ID of the image moderation task. |
| url | String | http://example.com/xxx.jpg | The URL of the moderated image. |
| storedUrl | String | http://aliyundoc.com | If you enable the feature of storing image evidence in Object Storage Service (OSS) buckets and the moderation task meets the configuration rule, the image is stored in your OSS bucket, and the corresponding HTTP URL is returned. |
| extras | JSONObject | {"hitLibInfo":[{"context":"Haokan","libCode":"2144002","libName":"Text pattern library a"}]} | The additional information.<br>If you specify *ad* in the scenes parameter to detect ad violations, the extras parameter can return the following content:<br>hitLibInfo: the information about the custom text library that the detected text in the moderated image hits. The value of this parameter is an array. For more information about the structure, see hitLibInfo. |
| results | JSONArray | | The return results. If HTTP status code 200 is returned after a successful call, the array in the return results contains one or more elements. Each element is a structure. For more information about the structure of each element, see result. |

result

| Parameter | Type | Example | Description |
|---|---|---|---|

| Parameter | Type | Example | Description |
|---|---|---|---|
| scene | String | porn | The moderation scenario of the moderated image, which you specify in the moderation request. Valid values:<br>- *porn*: pornography detection<br>- *terrorism*: terrorist content detection<br>- *ad*: ad violation detection<br>- *qrcode*: QR code detection<br>- *live*: undesirable scene detection<br>- *logo*: logo detection |
| label | String | sexy | The category of the moderation result of the moderated image. Valid values vary based on the specified moderation scenario.<br>- If the scenes parameter is set to *porn*, the valid values are:<br>  - *normal*: normal<br>  - *sexy*: sexy content<br>  - *porn*: pornographic content<br>- If the scenes parameter is set to *terrorism*, the valid values are:<br>  - *normal*: normal<br>  - *bloody*: bloody content<br>  - *explosion*: explosion and smoke<br>  - *outfit*: special costume<br>  - *logo*: logo<br>  - *weapon*: weapon<br>  - *politics*: political content<br>  - *violence*: violence<br>  - *crowd*: crowd<br>  - *parade*: parade<br>  - *carcrash*: car accident<br>  - *flag*: flag<br>  - *location*: landmark<br>  - *others*: other specified content<br>- If the scenes parameter is set to *ad*, the valid values are:<br>  - *normal*: normal<br>  - *ad*: other ads<br>  - *politics*: political content in text<br>  - *porn*: pornographic content in text<br>  - *abuse*: abuse in text<br>  - *terrorism*: terrorist content in text<br>  - *contraband*: prohibited content in text<br>  - *spam*: junk content in text<br>  - *npx*: illegal ad |

| Parameter | Type | Example | ○ *qrcode*: QR code<br>Description<br>○ *programCode*: mini program code |
|---|---|---|---|
| | | | • If the scenes parameter is set to *qrcode*, the valid values are:<br>○ *normal*: normal<br>○ *qrcode*: QR code<br>○ *programCode*: mini program code<br>• If the scenes parameter is set to *live*, the valid values are:<br>○ *normal*: normal<br>○ *meaningless*: no content in the image, such as black or white screen<br>○ *PIP*: small picture<br>○ *smoking*: smoking content<br>○ *drivelive*: live broadcasting in a running vehicle<br>• If the scenes parameter is set to *logo*, the valid values are:<br>○ *normal*: normal<br>○ *TV*: TV station logo<br>○ *trademark*: trademark |
| sublabel | String | porn | |
| suggestion | String | block | The recommended subsequent operation for you to perform. Valid values:<br>• *pass*: The moderation object does not require further actions.<br>• *review*: The moderation object contains suspected violations and requires human review.<br>• *block*: The moderation object contains violations. We recommend that you delete or block the object. |
| rate | Float | 91.54 | |
| frames | JSONArray | | The temporary access URL of each frame that is truncated from the moderated image because the image is too long. For more information about the structure, see frame. |
| hintWordsInfo | JSONArray | | The information about the term that the detected ad or illegal text in the moderated image hits. For more information about the structure, see hintWordsInfo.<br>**Note** This parameter is applicable only to ad violation detection.<br>Example:<br>`"hintWordsInfo":[{"context":"Sensitive words"}]` |

| Parameter | Type | Example | Description |
|---|---|---|---|
| qrcodeData | StringArray | ["http://example.com/01ZZOliO"] | The information about the text that is included in the detected QR code in the moderated image.<br>Note This parameter is applicable only to QR code detection. |
| qrcodeLocations | JSONArray | | The coordinate information about the detected QR code in the moderated image. For more information about the structure, see qrcodeLocation. |
| programCodeData | JSONArray | | The location information about the detected mini program code in the moderated image. For more information about the structure, see programCodeData.<br>Note This parameter is applicable only to mini program code detection. Make sure that mini program code detection is enabled. |
| logoData | JSONArray | | The information about the detected logo in the moderated image. For more information about the structure, see logoData.<br>Note This parameter is applicable only to logo detection. |
| sfaceData | JSONArray | | The information about the detected terrorist content in the moderated image. For more information about the structure, see sfaceData.<br>Note This parameter is applicable only to terrorist content detection. |
| ocrData | StringArray | xxxx | The information about the detected complete text in the moderated image.<br>Note By default, this parameter is not returned. If you want this parameter to be returned, submit a . |

frame

| Parameter | Type | Example | Description |
|---|---|---|---|
| rate | Float | 89.85 | |

| Parameter | Type | Example | Description |
|---|---|---|---|
| url | String | https://example.com/xxx-0.jpg | The temporary access URL of the truncated frame. The URL is valid for 5 minutes. |

programCodeData

| Parameter | Type | Example | Description |
|---|---|---|---|
| x | Float | 11.0 | The distance between the upper-left corner of the mini program code area and the y-axis, with the upper-left corner of the image being the coordinate origin. Unit: pixel. |
| y | Float | 0.0 | The distance between the upper-left corner of the mini program code area and the x-axis, with the upper-left corner of the image being the coordinate origin. Unit: pixel. |
| w | Float | 402.0 | The width of the mini program code area. Unit: pixel. |
| h | Float | 413.0 | The height of the mini program code area. Unit: pixel. |

logoData

| Parameter | Type | Example | Description |
|---|---|---|---|
| type | String | TV | The type of the detected logo. For example, a value of TV indicates the logo of a TV station. |
| name | String | xxx TV | The name of the detected logo. |
| x | Float | 140 | The distance between the upper-left corner of the logo area and the y-axis, with the upper-left corner of the image being the coordinate origin. Unit: pixel. |
| y | Float | 68 | The distance between the upper-left corner of the logo area and the x-axis, with the upper-left corner of the image being the coordinate origin. Unit: pixel. |
| w | Float | 106 | The width of the logo area. Unit: pixel. |
| h | Float | 106 | The height of the logo area. Unit: pixel. |

sfaceData

| Parameter | Type | Example | Description |
|---|---|---|---|
| x | Float | 49 | The distance between the upper-left corner of the face area and the y-axis, with the upper-left corner of the image being the coordinate origin. Unit: pixel. |

| Parameter | Type | Example | Description |
|---|---|---|---|
| y | Float | 39 | The distance between the upper-left corner of the face area and the x-axis, with the upper-left corner of the image being the coordinate origin. Unit: pixel. |
| w | Float | 97 | The width of the face area. Unit: pixel. |
| h | Float | 131 | The height of the face area. Unit: pixel. |
| faces | JSONArray | [{"id":"AliFace_0001234","name":"Hitme"name","rate":91.54}] | The information about the recognized face. The array contains the following parameters:<br>● name: the name of the recognized face. The value is a string.<br>● rate: The value is a floating-point number.<br>● id: the ID of the recognized face. The value is a string. |

hitLibInfo

| Parameter | Type | Example | Description |
|---|---|---|---|
| context | String | xxxx | The custom text that the detected text hits. |
| libCode | String | 123456 | The code of the library that contains the custom text hit by the detected text. |
| libName | String | abc | The name of the library that contains the custom text hit by the detected text. |

hintWordsInfo

| Parameter | Type | Example | Description |
|---|---|---|---|
| context | String | xxxx | The term that the detected text hits. |

qrcodeLocation

| Parameter | Type | Example | Description |
|---|---|---|---|
| x | Float | 11.0 | The distance between the upper-left corner of the QR code area and the y-axis, with the upper-left corner of the image being the coordinate origin. Unit: pixel. |
| y | Float | 0.0 | The distance between the upper-left corner of the QR code area and the x-axis, with the upper-left corner of the image being the coordinate origin. Unit: pixel. |
| w | Float | 402.0 | The width of the QR code area. Unit: pixel. |
| h | Float | 413.0 | The height of the QR code area. Unit: pixel. |

| Parameter | Type | Example | Description |
|---|---|---|---|
| qrcode | String | http://xxx | The URL in the detected QR code. |

## Examples

Sample requests

```
{
    "scenes": [
        "porn",
        "terrorism",
        "ad",
        "live",
        "qrcode",
        "logo"
    ],
    "tasks": [
        {
            "dataId": "uuid-xxxx-xxxx-1234",
            "url": "http://example.com/xxx.jpg"
        }
    ]
}
```

Sample success responses

```
{
     "msg": "OK",
     "code": 200,
     "data": [
          {
               "msg": "OK",
               "code": 200,
               "dataId": "uuid-xxxx-xxxx-1234",
               "extras": {
            },
               "results": [
                    {
                         "rate": 99.63,
                         "suggestion": "block",
                         "label": "sexy",
                         "scene": "porn"
                },
                    {
                         "label": "politics",
                         "rate": 91.54,
                         "scene": "terrorism",
                         "sfaceData": [
                              {
                                   "faces": [
                                        {
```

```
                             "id": "AliFace_0001234",
                             "name": "Hit name",
                             "rate": 91.54
                a}
            ],
                   "h": 131,
                   "w": 97,
                   "x": 49,
                   "y": 39
        }
    ],
         "suggestion": "block"
},
    {
         "extras": {
              "qrcodes": "http://example.com/0.ZZOliO",
              "npx": "72.01",
              "hitCustomLibCode": "8012345000",
              "hitCustomLibName": "Name of the custom image library",
              "hitLibInfo": [
                   {
                        "context": "Hit text",
                        "libCode": "123456",
                        "libName": "Name of the custom text library"
            }
        ]
    },
         "programCodeData": [
              {
                   "w": 402.0,
                   "h": 413.0,
                   "x": 11.0,
                   "y": 0.0
        }
    ],
         "frames": [
              {
                   "rate": 89.85,
                   "url": "http://example.com/xxx-0.jpg"
        },
              {
                   "rate": 68.06,
                   "url": "http://example.com/xxx-1.jpg"
        }
    ],
         "rate": 99.91,
         "suggestion": "block",
         "label": "ad",
         "scene": "ad"
},
    {
         "rate": 99.91,
         "suggestion": "block",
         "label": "drug",
         "scene": "live"
```

```
                          "scene": "live"
                }, 
                    {
                         "qrcodeData": [
                              "http://example.com/01ZZOliO"
                    ], 
                         "rate": 99.91, 
                         "suggestion": "review", 
                         "label": "qrcode", 
                         "scene": "qrcode"
                }, 
                    {
                         "logoData": [
                              {
                                   "name": "xxx TV", 
                                   "type": "TV", 
                                   "x": 140, 
                                   "y": 68, 
                                   "w": 106, 
                                   "h": 106
                        }
                    ], 
                         "rate": 99.9, 
                         "suggestion": "block", 
                         "label": "TV", 
                         "scene": "logo"
                }
            ], 
               "taskId": "img4wlJcb7p4wH4lAP3111111-123456", 
               "url": "http://example.com/xxx.jpg"
        }
    ], 
     "requestId": "69B41AE8-1234-1234-1234-12D395695D2D"
}
```

## 5.1.2. Asynchronous moderation

This topic describes the /green/image/asyncscan operation that you can call to asynchronously moderate images. You can call this operation to moderate images for risky or illegal content. This operation is applicable to the following scenarios: pornography detection, terrorist content detection, ad violation detection, QR code detection, undesirable scene detection, and logo detection.

### Description about the /green/image/asyncscan operation

Operation: /green/image/asyncscan

You can call this operation to submit asynchronous image moderation tasks.

- **Billing method:**
- **Response timeout:**
- **Return results:**

- Enable callback notification to obtain moderation results: When you submit asynchronous moderation tasks, you can specify a callback URL for receiving moderation results in the callback parameter of the moderation request. For more information about the callback parameter, see Request parameters.
- Poll moderation results at regular intervals: You do not need to set the callback parameter when you submit asynchronous moderation tasks. After you submit the tasks, you can call the result query operation to query moderation results. For more information about the operation, see Description about the /green/image/results operation.

- Limits on images:

## Request parameters

| Parameter | Type | Required | Example | Description |
|---|---|---|---|---|
| bizType | String | No | default | |
| scenes | String Array | Yes | ["porn"] | The moderation scenario. Valid values:<br>• *porn*: pornography detection<br>• *terrorism*: terrorist content detection<br>• *ad*: ad violation detection<br>• *qrcode*: QR code detection<br>• *live*: undesirable scene detection<br>• *logo*: logo detection<br>You can specify multiple scenarios. For example, you can specify both porn and terrorism in the `scenes` parameter to detect pornographic and terrorist content.<br>**Note** If your moderation occurs in multiple scenarios at a time, you are charged the cumulative fee of all scenarios. The fee of each scenario equals the number of images that are moderated in the scenario multiplied by the unit price of the scenario. |
| callback | String | No | http://example.com | |
| seed | String | No | abc_123 | |

| Parameter | Type | Required | Example | Description |
|---|---|---|---|---|
| cryptType | String | No | SHA256 | The encryption algorithm used to encrypt the callback content when you enable callback notification. Content Moderation encrypts the returned string based on the encryption algorithm that you specify and then sends the encrypted string to the callback URL. The returned string is in the `UID + Seed + Content` format. Valid values:<br>- **SHA256**: The HMAC-SHA256 encryption algorithm is used. This is the default value.<br>- **SM3**: The HMAC-SM3 encryption algorithm is used and a hexadecimal string is returned. The string consists of lowercase letters and digits.<br>For example, **66c7f0f462eeedd9d1f2d46bdc10e4e24167c4875cf2f7a2297da02b8f4ba8e0** is returned after you encrypt **abc** by using the HMAC-SM3 encryption algorithm. |
| offline | Boolean | No | false | The moderation mode. Valid values:<br>- true: indicates the nearline moderation mode. In nearline moderation mode, the moderation tasks you submit may not be processed in real time, but can be queued for processing and will start within 24 hours.<br>- false: indicates the real-time moderation mode. This is the default value. In this mode, Content Moderation rejects moderation requests that exceed the concurrency limit. |
| tasks | JSONArray | Yes | | The list of moderation objects. Each element in the JSON array is a moderation task structure. The JSON array can contain a maximum of 100 elements. In other words, you can submit a maximum of 100 images at a time. To submit 100 images at a time, you must raise the relevant concurrency limit to a number greater than 100. For more information about the structure of each element, see task. |

task

| Parameter | Type | Required | Example | Description |
|---|---|---|---|---|
| clientInfo | JSONObject | No | null | |
| dataId | String | No | test4lNSMdggA0c56MMvfYoh4e-1mwxpx | |
| url | String | Yes | https://img.alicdn.com/tfs/TB1urBOQFXXXXbMXFXXXXXXXXXX-1442-257.png | The URL of the image to be moderated. |
| extras | JSONObject | No | | The extra parameters that you can set for calling the operation. This parameter is not required for image moderation. |

| Parameter | Type | Required | Example | Description |
|---|---|---|---|---|
| interval | Integer | No | 1 | The interval between two frames that are consecutively captured. This parameter is dedicated for GIF or long image moderation.<br>• A GIF image can be regarded as an array of frames. One frame is captured for moderation from every n frames. The n is specified by the interval parameter. The system captures frames from GIF images only when this parameter is specified.<br>• Long images can be in portrait or horizontal mode.<br>○ To moderate a long portrait image, in which the height is greater than 400 pixels and the ratio of height to width is greater than 2.5, you can divide the height by the width and round up the result to the nearest integer as the total number of frames.<br>○ To moderate a long horizontal image, in which the width is greater than 400 pixels and the ratio of width to height is greater than 2.5, you can divide the width by the height and round up the result to the nearest integer as the total number of frames.<br>By default, only the first frame of a GIF image or a long image is moderated. You can use the interval parameter to specify the interval between two frames that the system consecutively captures. This helps reduce moderation costs.<br>**Note** The interval and maxFrames parameters must be used in pairs. For example, the interval parameter is set to 2 and the maxFrames parameter is set to 20 for moderating a GIF image or a long image. In this example, one out of every two frames is moderated and a maximum of 20 frames are moderated. The fee is calculated based on the actual number of moderated frames. |
| maxFrames | Integer | No | 20 | |

## Response parameters

| Parameter | Type | Example | Description |
|---|---|---|---|
| code | Integer | 200 | |
| msg | String | OK | The message that is returned for the request. |
| dataId | String | test4lNSMdggA0c56MMvfYoh4e-1mwxpx | |
| taskId | String | fdd25f95-4892-4d6b-aca9-7939bc6e9baa-1486198766695 | The ID of the moderation task. |
| url | String | https://example.com/tfs/TB1urBOQFXXXXbMXFXXXXXXXXXX-1442-257.png | The URL of the moderation object. |

## Examples

Sample requests

```
{
    "scenes": [
        "porn"
    ],
    "tasks": [
        {
            "dataId": "test4lNSMdggA0c56MMvfYoh4e-1mwxpx",
            "url": "https://example.com/tfs/TB1urBOQFXXXXbMXFXXXXXXXXXX-1442-257.png"
        }
    ]
}
```

Sample success responses

```
{
    "code": 200,
    "msg": "OK",
    "requestId": "95AD868A-F5D2-4AEA-96D4-E0273B8E074C",
    "data": [
        {
            "code": 200,
            "msg": "OK",
            "dataId": "test41NSMdggA0c56MMvfYoh4e-1mwxpx",
            "taskId": "fdd25f95-4892-4d6b-aca9-7939bc6e9baa-1486198766695",
            "url": "https://example.com/tfs/TB1urBOQFXXXXbMXFXXXXXXXXXX-1442-257.png"
        }
    ]
}
```

## Description about the /green/image/results operation

Operation: /green/image/results

You can call this operation to query asynchronous image moderation results.

- **Billing method:**
- **Response timeout:**

  We recommend that you query moderation results at least 30s after you send an asynchronous moderation request and obtain the value of the taskId parameter.

> **Note** If you set the offline parameter to **true**, the validity period for the value of the taskId parameter is 24 hours. If you set the offline parameter to **false**, the validity period for the value of the taskId parameter is an hour. The default value of the offline parameter is **false**.

## Request parameters

| Parameter | Type | Required | Example | Description |
|---|---|---|---|---|
| body | JSONArray | Yes | ["fdd25f95-4892-4d6b-aca9-7939bc6e9baa-1486198766695"] | |

## Response parameters

| Parameter | Type | Example | Description |
|---|---|---|---|
| code | Integer | 200 | |
| msg | String | OK | The message that is returned for the request. |

| Parameter | Type | Example | Description |
|---|---|---|---|
| dataId | String | uuid-xxxx-xxx-1234 | |
| taskId | String | img4wlJcb7p4wH4lAP3111111-123456 | The ID of the moderation task. |
| url | String | http://example.com/xxx.jpg | The URL of the moderation object. |
| storedUrl | String | http://aliyundoc.com | If you enable the feature of storing image evidence in Object Storage Service (OSS) buckets and the moderation task meets the configuration rule, the image is stored in your OSS bucket, and the corresponding HTTP URL is returned. |
| extras | JSONObject | {"hitLibInfo":[{"context":"Haokan","libCode":"2144002","libName":"Text pattern library a"}]} | The additional information.<br>If you specify *ad* in the scenes parameter to detect ad violations, the extras parameter can return the following content:<br>hitLibInfo: the information about the custom text library that the detected text in the moderated image hits. This parameter is of the array type. For more information about the structure, see hitLibInfo. |
| results | JSONArray | | The return results. If HTTP status code 200 is returned after a successful call, the array in the return results contains one or more elements. Each element is a structure. For more information about the structure, see result. |

result

| Parameter | Type | Example | Description |
|---|---|---|---|
| scene | String | terrorism | The moderation scenario of the moderated image, which you specify in the moderation request. Valid values: |
| | | | The category of the moderation result. Valid values vary based on the specified moderation scenario.<br>● If the scenes parameter is set to *porn*, the valid values are:<br>○ *normal*: normal<br>○ *sexy*: sexy content<br>○ *porn*: pornographic content<br>● If the scenes parameter is set to *terrorism*, the valid values are:<br>○ *normal*: normal |

| Parameter | Type | Example | Description |
|---|---|---|---|
| label | String | sexy | ○ *bloody*: bloody content<br>○ *explosion*: explosion and smoke<br>○ *outfit*: special costume<br>○ *logo*: logo<br>○ *weapon*: weapon<br>○ *politics*: political content<br>○ *violence*: violence<br>○ *crowd*: crowd<br>○ *parade*: parade<br>○ *carcrash*: car accident<br>○ *flag*: flag<br>○ *location*: landmark<br>○ *others*: other specified content<br>• If the scenes parameter is set to *ad*, the valid values are:<br>○ *normal*: normal<br>○ *ad*: other ads<br>○ *politics*: political content in text<br>○ *porn*: pornographic content in text<br>○ *abuse*: abuse in text<br>○ *terrorism*: terrorist content in text<br>○ *contraband*: prohibited content in text<br>○ *spam*: junk content in text<br>○ *npx*: illegal ad<br>○ *qrcode*: QR code<br>○ *programCode*: mini program code<br>• If the scenes parameter is set to *qrcode*, the valid values are:<br>○ *normal*: normal<br>○ *qrcode*: QR code<br>○ *programCode*: mini program code<br>• If the scenes parameter is set to *live*, the valid values are:<br>○ *normal*: normal<br>○ *meaningless*: no content in the image, such as black or white screen<br>○ *PIP*: small picture<br>○ *smoking*: smoking content<br>○ *drivelive*: live broadcasting in a running vehicle<br>• If the scenes parameter is set to *logo*, the valid values are:<br>○ *normal*: normal<br>○ *TV*: TV station logo<br>○ *trademark*: trademark |

| Parameter | Type | Example | Description |
|---|---|---|---|
| sublabel | String | | |
| suggestion | String | block | The recommended subsequent operation for you to perform. Valid values:<br>- *pass*: The moderation object does not require further actions.<br>- *review*: The moderation object contains suspected violations and requires human review.<br>- *block*: The moderation object contains violations. We recommend that you delete or block the object. |
| rate | Float | 91.54 | |
| frames | JSONArray | | The temporary access URL of each frame that is truncated from the moderated image because the image is too long. For more information about the structure, see frame. |
| hintWordsInfo | JSONArray | {"hintWordsInfo":[{"context":"Sensitive words"}]} | The information about the term that the detected ad in the moderated image hits. This parameter is of the array type. For more information about the structure, see hintWordsInfo.<br>Note This parameter is applicable only to ad violation detection. |
| qrcodeData | StringArray | ["http://example.com/01ZZOliO"] | The information about the text that is included in the detected QR code in the moderated image.<br>Note This parameter is applicable only to QR code detection. |
| qrcodeLocations | JSONArray | | The coordinate information about the detected QR code in the moderated image. For more information about the structure, see qrcodeLocation. |
| programCodeData | JSONArray | | The location information about the detected mini program code in the moderated image. For more information about the structure, see programCodeData.<br>Note This parameter is applicable only to mini program code detection. Make sure that mini program code detection is enabled. |

| Parameter | Type | Example | Description |
|---|---|---|---|
| logoData | JSONArray | | The information about the detected logo in the moderated image. For more information about the structure, see logoData.<br>Note This parameter is applicable only to logo detection. |
| sfaceData | JSONArray | | The information about the detected terrorist content in the moderated image. For more information about the structure, see sfaceData.<br>Note This parameter is applicable only to terrorist content detection. |
| ocrData | StringArray | xxxx | The information about the detected complete text in the moderated image.<br>Note By default, this parameter is not returned. If you want this parameter to be returned, submit a . |

frame

| Parameter | Type | Example | Description |
|---|---|---|---|
| rate | Float | 89.85 | |
| url | String | http://example.com/xxx-0.jpg | The temporary access URL of the truncated frame. The URL is valid for 5 minutes. |

## Examples

Sample requests

```
[
    "fdd25f95-4892-4d6b-aca9-7939bc6e9baa-1486198766695"
]
```

Sample success responses

```
{
     "msg": "OK",
     "code": 200,
     "data": [
          {
               "msg": "OK",
               "code": 200,
               "dataId": "test4lNSMdggA0c56MMvfYoh4e-1mwxpx",
               "extras": {
            },
               "results": [
                    {
                         "rate": 99.63,
                         "suggestion": "block",
                         "label": "sexy",
                         "scene": "porn"
                },
                    {
                         "label": "politics",
                         "rate": 91.54,
                         "scene": "terrorism",
                         "sfaceData": [
                              {
                                   "faces": [
                                        {
                                             "id": "AliFace_0001234",
                                             "name": "Hit name",
                                             "rate": 91.54
                                }
                            ],
                                   "h": 131,
                                   "w": 97,
                                   "x": 49,
                                   "y": 39
                        }
                    ],
                         "suggestion": "block"
                },
                    {
                         "extras": {
                              "qrcodes": "http://example.com/0.ZZOliO",
                              "npx": "72.01",
                              "hitCustomLibCode": "8012345000",
                              "hitCustomLibName": "Name of the custom image library",
                              "hitLibInfo": [
                                   {
                                        "context": "Hit text",
                                        "libCode": "123456",
                                        "libName": "Name of the text library"
                            }
                        ]
                    },
                         "programCodeData": [
                              {
                                   "w": 402.0,
                                   "h": 413.0,
                                   "x": 11.0,
                                   "y": 0.0
                        }
                    ],
                         "frames": [
                              {
                                   "rate": 89.85,
                                   "url": "http://example.com/xxx-0.jpg"
                        },
                              {
                                   "rate": 68.06,
                                   "url": "http://example.com/xxx-1.jpg"
                        }
                    ],
                         "rate": 99.91,
                         "suggestion": "block",
                         "label": "ad",
                         "scene": "ad"
                },
                    {
                         "rate": 99.91,
                         "suggestion": "block",
                         "label": "drug",
                         "scene": "live"
                },
                    {
                         "qrcodeData": [
                              "http://example.com/01ZZOliO"
                    ],
                         "rate": 99.91,
                         "suggestion": "review",
                         "label": "qrcode",
                         "scene": "qrcode"
                },
                    {
                         "logoData": [
                              {
                                   "name": "xxx TV",
                                   "type": "TV",
                                   "x": 140,
                                   "y": 68,
                                   "w": 106,
                                   "h": 106
                        }
                    ],
                         "rate": 99.9,
                         "suggestion": "block",
                         "label": "TV",
                         "scene": "logo"
                }
            ],
               "taskId": "fdd25f95-4892-4d6b-aca9-7939bc6e9baa-1486198766695",
               "url": "https://example.com/tfs/TB1urBOQFXXXXbMXFXXXXXXXXXX-1442-257.png"
        }
```

```
    ],
     "requestId": "69B41AE8-1234-1234-1234-12D395695D2D"
}
```

## 5.1.3. Give feedback on moderation results

If you find that image moderation results returned by Alibaba Cloud Content Moderation do not meet your expectations, you can call an operation to provide feedback. Your suggestion will take effect the next time you moderate the same image.

### Description

Operation: /green/image/feedback

You can call this operation to give feedback on the result of an image moderation task and provide your expected result.

> **Note** You are not charged for calling this operation.

### Request parameters

For more information about the common request parameters that must be included in all Content Moderation API requests, see Common request parameters.

The request body is a JSON structure. The following table describes the parameters that the JSON structure contains.

| Parameter | Type | Required | Description |
|---|---|---|---|
| taskId | String | No | The ID returned by the Content Moderation server, which uniquely identifies the moderation task. |
| suggestion | String | No | The moderation result you expect to return. If you specify this parameter, you must specify the scenes parameter. Valid values:<br>• *pass*: The image is normal.<br>**Note** The image whose moderation result is *pass* can be automatically added to the whitelist of the feedback-based image library in the Alibaba Cloud Content Moderation console.<br>• *block*: The image contains violations. |
| url | String | No | The URL for adding the moderated image to the feedback-based image library in the Alibaba Cloud Content Moderation console. To add the image to the feedback-based image library, you must specify this parameter. |

| Parameter | Type | Required | Description |
|---|---|---|---|
| scenes | String array | No | The scenario of image moderation. Valid values:<br>• *porn*: pornography detection<br>• *terrorism*: terrorist content detection<br>• *ad*: ad violation detection<br>**Note** You can specify multiple moderation scenarios. For example, you can specify both porn and terrorism in the `scenes` parameter to indicate pornography detection and terrorist content detection scenarios. |
| note | String | No | The description of the moderated image. |
| label | String | No | The expected category of moderation results for the moderated image in the specified moderation scenario. For more information about valid values, see Parameters scenes and label. |

## Response parameters

The data parameter in the response body is empty. For more information about common response parameters that this operation returns, see Common response parameters.

## Examples

### Sample requests

```
{
  "suggestion": "block",
  "scenes":[
    "ad",
    "terrorism"
  ],
  "url": "http://xxx.xxx.xxx.com"
}
```

### Sample success responses

```
{
  "code": 200,
  "msg": "OK",
  "requestId": "EE5A1189-4D7B-4C24-AD78-4C1FAA3E7A0C"
}
```

# 5.2. Video moderation

## 5.2.1. Asynchronous moderation

This topic describes the operation that you can call to asynchronously moderate videos. You can call this operation to moderate videos for risky or illegal content. This operation is applicable to the following scenarios: pornography detection, terrorist content detection, ad violation detection, undesirable scene detection, logo detection, and audio anti-spam.

## Description about the /green/video/asyncscan operation

Operation: /green/video/asyncscan

You can call this operation to submit asynchronous video moderation tasks.

- **Billing rule:**

  If your moderation occurs in multiple scenarios at a time, you are charged the cumulative fee of all scenarios. The fee of each scenario equals `the number of video frames that are moderated in the scenario multiplied by the unit price of the scenario`. If you moderate the audio in a video at the same time, you are charged an extra fee for audio anti-spam. The extra fee equals the video duration multiplied by the unit price of audio anti-spam.

- **Moderated object:**

  You can call this operation to moderate videos or video streams. You can submit a sequence of frames that are captured from a video or submit a video URL to specify the video to be moderated.

- **Return results:**

  - Enable callback notification to obtain moderation results: When you submit asynchronous moderation tasks, you can specify a callback URL for receiving moderation results in the callback parameter of the moderation request. For more information about the callback parameter, see Request parameters.
  - Poll moderation results at regular intervals: You do not need to set the callback parameter when you submit asynchronous moderation tasks. After you submit the tasks, you can call the /green/video/results operation to query moderation results. For more information about the operation, see Description about the /green/video/results operation.

- **Limits on videos:**

## QPS limit

You can call this operation up to 50 times per second per account. The system supports a maximum of 20 concurrent moderation tasks. Requests that exceed this limit are dropped and you will experience service interruptions. We recommend that you take note of this limit when you call this operation.

## Request parameters

| Parameter | Type | Required | Example | Description |
|---|---|---|---|---|
| bizType | String | No | default | |
| live | Boolean | No | false | Specifies whether to moderate live streams. Valid values:<br>• **false**: moderates videos. This is the default value.<br>• **true**: moderates live streams. |

| Parameter | Type | Required | Example | Description |
|---|---|---|---|---|
| offline | Boolean | No | false | Specifies whether to enable the nearline moderation mode. Valid values:<br>- **false**: enables the real-time moderation mode. In this mode, Content Moderation rejects moderation requests that exceed the concurrency limit. This is the default value.<br>- **true**: enables the nearline moderation mode. In this mode, the moderation tasks you submit may not be processed in real time, but can be queued for processing and will start within 24 hours.<br>? **Note** This parameter is applicable only to video moderation. This parameter is not required for video stream moderation. |
| scenes | String Array | Yes | ["porn"] | The video moderation scenario. Valid values:<br>- **porn**: pornography detection<br>- **terrorism**: terrorist content detection<br>- **live**: undesirable scene detection<br>- **logo**: logo detection<br>- **ad**: ad violation detection |
| audioScenes | String Array | No | ["antispam"] | The audio moderation scenario. Set the value to *antispam*.<br>If you do not set this parameter, Content Moderation moderates only the images in videos. If you set this parameter, Content Moderation also moderates the audio in a video during video moderation.<br>? **Note** To moderate the audio in a video, you must set the url parameter in the task parameter to submit the URL of the video or video stream. Audio moderation is inapplicable if you set the frames parameter in the task parameter to submit frames that are captured from a video. |

| Parameter | Type | Required | Example | Description |
|---|---|---|---|---|
| callback | String | No | http://www.aliyundoc.com | |
| seed | String | No | abc**** | |
| | | | | |
| tasks | JSONArray | Yes | | The list of moderated objects. The JSON array can contain one or more elements. Each element is a structure. The JSON array can contain up to 100 elements. In other words, you can submit up to 100 moderated objects at a time. To submit 100 moderated objects at a time, you must raise the relevant concurrency limit to a number greater than 100. For more information about the structure of each element, see task. |

task

| Parameter | Type | Required | Example | Description |
|---|---|---|---|---|
| clientInfo | JSONObject | No | null | |
| dataId | String | No | videoId**** | |
| liveId | String | No | liveId**** | The ID of the video live stream.<br>This parameter is used to prevent repeated moderation for live stream tasks. If you specify this parameter, Content Moderation checks whether a live stream task exists for the same video live stream based on the ID of the Alibaba Cloud account and the bizType and liveId parameters. If the live stream task exists, the ID of the moderation task is returned, and no new task is initiated. |

| Parameter | Type | Required | Example | Description |
|---|---|---|---|---|
| url | String | No | http://www.aliyundoc.com/a.flv | The URL of the video to be moderated.<br>**Note** You must set one of the frames and url parameters. If you set the url parameter, you are charged for the video moderation task based on the video URL that is submitted. |
| frames | JSONArray | No |  | The information about frames that are captured from the video to be moderated. Each element in the JSON array of the frames parameter is a structure. For more information about the structure of each element, see frame.<br>**Note** You must set one of the frames and url parameters. If you set the frames parameter, you are charged for the video moderation task based on the captured frames that are submitted. |
| framePrefix | String | No | http://www.aliyundoc.com/video/ | The prefix of the URL of a captured frame, which is used with `frame.url` to form the complete URL of a captured frame. The complete URL of a captured frame is in the format of `framePrefix + frame.url`. |

| Parameter | Type | Required | Example | Description |
|---|---|---|---|---|
| interval | Integer | No | 1 | The interval for capturing frames from the video. Unit: seconds. Valid values: 1 to 600. Default value: 1. |
| maxFrames | Integer | No | 200 | The maximum number of frames that can be captured from the video. Valid values: 5 to 3600. Default value: 200. If you need to set this parameter to a greater value, submit a .<br>Note<br>- This parameter is valid only when you set the live parameter to *false* to moderate videos. If you set the live parameter to *true* to moderate live streams, the number of frames that can be captured from live streams is not limited.<br>- If you use an Object Storage Service (OSS) URL that starts with `oss://` as the source URL of a video and authorize Content Moderation |

| Parameter | Type | Required | Example | Description |
|---|---|---|---|---|
| | | | | to access ApsaraVideo Media Processing (MPS), you can capture a maximum of 20,000 frames from the video. You are not charged extra fees when you use this method. For more information about how to authorize Content Moderation to access MPS, see Authorize a role to access ApsaraVideo for Media Processing. |

frame

| Parameter | Type | Required | Example | Description |
|---|---|---|---|---|
| url | String | No | http://www.aliyundoc.com/0B860000586C0A0300038A0460000 | The URL of the captured frame, which is used with `framePrefix` to form the complete URL of a captured frame. The complete URL of a captured frame is in the format of `framePrefix + frame.url` . |
| offset | Integer | No | 10 | The interval between the start of the video and the captured frame. Unit: seconds. |

## Response parameters

| Parameter | Type | Example | Description |
|---|---|---|---|
| taskId | String | taskId**** | The ID of the moderation task. |

| Parameter | Type | Example | Description |
|---|---|---|---|
| dataId | String | videoId**** | |

## Examples

Sample requests

- Moderate frames that are captured from a video

```
http(s)://[Endpoint]/green/video/asyncscan
&<Common request parameters>
{
    "scenes": [
        "porn"
    ],
    "tasks": [
        {
            "dataId": "videoId****",
            "frames": [
                {
                    "offset": 10,
                    "url": "http://www.aliyundoc.com/0B860000586C0A0300038A0460000"
                },
                {
                    "offset": 20,
                    "url": "http://www.aliyundoc.com/0B860000586C0A0300038A0460001"
                },
                {
                    "offset": 30,
                    "url": "http://www.aliyundoc.com/0B860000586C0A0300038A0460002"
                },
                {
                    "offset": 40,
                    "url": "http://www.aliyundoc.com/0B860000586C0A0300038A0460003"
                },
                {
                    "offset": 50,
                    "url": "http://www.aliyundoc.com/0B860000586C0A0300038A0460003"
                },
                {
                    "offset": 60,
                    "url": "http://www.aliyundoc.com/0B860000586C0A0300038A046000x"
                }
            ]
        }
    ]
}
```

- Moderate a video

```
http(s)://[Endpoint]/green/video/asyncscan
&<Common request parameters>
{
     "scenes": [
        "porn"
    ],
     "audioScenes": [
        "antispam"
    ],
     "tasks": [
          {
               "dataId": "videoId****",
               "url": "http://www.aliyundoc.com/a.mp4",
               "interval": 1,
               "maxFrames": 200
        }
    ]
}
```

- Moderate a live stream

```
http(s)://[Endpoint]/green/video/asyncscan
&<Common request parameters>
{
    "scenes": [
        "porn"
    ],
    "live": true,
    "tasks": [
        {
            "dataId": "videoId****",
            "url": "http://www.aliyundoc.com/a.flv",
            "interval": 1,
            "maxFrames": 200
        }
    ]
}
```

Sample success responses

```
{
    "code": 200,
    "msg": "OK",
    "requestId": "requestID****",
    "data": [
        {
            "dataId": "videoId****",
            "taskId": "taskId****"
        }
    ]
}
```

## Description about the /green/video/results operation

Operation: /green/video/results

You can call this operation to query asynchronous video moderation results.

- **Billing rule:**
- **Response timeout:**

## Request parameters

| Parameter | Type | Required | Example | Description |
|---|---|---|---|---|
| body | JSONArray | Yes | ["taskId****","taskId****"] | |

## Response parameters

| Parameter | Type | Example | Description |
|---|---|---|---|
| code | Integer | 200 | |
| msg | String | OK | The message that is returned for the request. |
| dataId | String | videoId**** | |
| taskId | String | taskId**** | The ID of the moderation task. |
| results | JSONArray | | The return results. If HTTP status code 200 is returned after a successful call, the array in the return results contains one or more elements. Each element is a structure. For more information about the structure, see result.<br>? **Note** In the scenario of moderating a video stream, HTTP status code 280 indicates that the moderation task is in progress, and HTTP status code 200 indicates that the moderation task is complete. If the moderation task is in progress, the returned moderation results contain all the issues that Content Moderation has detected in the task. |
| audioScanResults | JSONObject | | The audio moderation results. For more information about the structure, see audioScanResult. |

result

| Parameter | Type | Example | Description |
|---|---|---|---|

| Parameter | Type | Example | Description |
|---|---|---|---|
| scene | String | porn | The moderation scenario of the moderated video, which you specify in the moderation request. Valid values:<br>- **porn**: pornography detection<br>- **terrorism**: terrorist content detection<br>- **live**: undesirable scene detection<br>- **logo**: logo detection<br>- **ad**: ad violation detection |
| label | String | porn | The category of the moderation result of the moderated video. Valid values vary based on the specified moderation scenario. |
| sublabel | String | porn | |
| suggestion | String | block | The recommended subsequent operation. Valid values:<br>- *pass*: The moderated object does not require further actions.<br>- *review*: The moderated object contains suspected violations and requires human review.<br>- *block*: The moderated object contains violations. We recommend that you delete or block the object. |
| rate | Float | 99.2 | |
| frames | JSONArray | | The moderation results of captured frames. For more information about the structure, see frame. |
| extras | JSONObject | {"hitLibInfo":[{"context":"Haokan","libCode":"2144002","libName":"Text pattern library a"}]} | The additional information.<br>If you specify *ad* in the scenes parameter to detect ad violations, the extras parameter can return the following content:<br>hitLibInfo: the information about the custom text library that the detected text in the moderated video hits. For more information about the structure, see hitLibInfo. |
| hintWordsInfo | JSONArray | | The information about the term that the detected ad or illegal text in the moderated video hits. For more information about the structure, see hintWordsInfo.<br>**Note** This parameter is applicable only to ad violation detection. |

| Parameter | Type | Example | Description |
|---|---|---|---|
| logoData | JSONArray | | The information about the detected logo in the moderated video. For more information about the structure, see logoData.<br>**Note** This parameter is applicable only to logo detection. |
| sfaceData | JSONArray | | The information about the detected terrorist content in the moderated video. For more information about the structure, see sfaceData.<br>**Note** This parameter is applicable only to terrorist content detection. |

frame

| Parameter | Type | Example | Description |
|---|---|---|---|
| url | String | http://www.aliyundoc.com/0B860000586C0A0 | The URL of the captured frame. |
| offset | Integer | 50 | The interval between the start of the video and the captured frame. Unit: seconds. |
| | | | The category of the moderation result of the captured frame. Valid values vary based on the specified moderation scenario.<br>• If the scenes parameter is set to porn, the valid values are:<br>○ normal: normal<br>○ sexy: sexy content<br>○ porn: pornographic content<br>• If the scenes parameter is set to terrorism, the valid values are:<br>○ normal: normal<br>○ bloody: bloody content<br>○ explosion: explosion and smoke<br>○ outfit: special costume<br>○ logo: logo<br>○ weapon: weapon<br>○ politics: political content<br>○ violence: violence<br>○ crowd: crowd |

| Parameter | Type | Example | Description |
|---|---|---|---|
| label | String | porn | ○ parade: parade<br>○ carcrash: car accident<br>○ flag: flag<br>○ location: landmark<br>○ others: other specified content<br>• If the scenes parameter is set to ad, the valid values are:<br>○ normal: normal<br>○ politics: political content in text<br>○ porn: pornographic content in text<br>○ abuse: abuse in text<br>○ terrorism: terrorist content in text<br>○ contraband: prohibited content in text<br>○ spam: junk content in text<br>○ npx: illegal ad<br>○ qrcode: QR code<br>○ programCode: mini program code<br>○ ad: other ads<br>• If the scenes parameter is set to live, the valid values are:<br>○ normal: normal<br>○ meaningless: meaningless, such as black or white screen<br>○ PIP: small picture<br>○ smoking: smoking content<br>○ drivelive: live broadcasting in a running vehicle<br>• If the scenes parameter is set to logo, the valid values are:<br>○ normal: normal<br>○ TV: TV station logo<br>○ trademark: trademark |
| rate | Float | 99.1 | |

audioScanResult

| Parameter | Type | Example | Description |
|---|---|---|---|
| scene | String | antispam | The audio moderation scenario of the moderated video. The value is fixed to *antispam*. |

| Parameter | Type | Example | Description |
|---|---|---|---|
| label | String | customized | The category of the audio moderation result for the moderated video. Valid values vary based on the specified moderation scenario.<br>- *normal*: normal<br>- *spam*: junk content<br>- *ad*: ad<br>- *politics*: political content<br>- *terrorism*: terrorist content<br>- *abuse*: abuse<br>- *porn*: pornographic content<br>- *flood*: excessive junk content<br>- *contraband*: prohibited content<br>- *customized*: custom content, such as a custom term |
| suggestion | String | block | The recommended subsequent operation. Valid values:<br>- *pass*: The moderated object does not require further actions.<br>- *review*: The moderated object contains suspected violations and requires human review.<br>- *block*: The moderated object contains violations. We recommend that you delete or block the object. |
| rate | Float | 99.91 | |
| details | JSONArray | | The details about text in the moderated audio. The JSON array contains one or more elements. Each element corresponds to a text entry. For more information, see detail. |

### detail

| Parameter | Type | Example | Description |
|---|---|---|---|
| startTime | Integer | 24 | The start time of the text entry. Unit: seconds. |
| endTime | Integer | 60 | The end time of the text entry. Unit: seconds. |
| text | String | Computer | The content of the text entry that is converted from the audio. |

| Parameter | Type | Example | Description |
|---|---|---|---|
| label | String | normal | The category of the moderation result of the text entry. Valid values:<br>• *normal*: normal<br>• *spam*: junk content<br>• *ad*: ad<br>• *politics*: political content<br>• *terrorism*: terrorist content<br>• *abuse*: abuse<br>• *porn*: pornographic content<br>• *flood*: excessive junk content<br>• *contraband*: prohibited content<br>• *customized*: custom content, such as a custom term |
| keyword | String | Enable | The custom term that the text entry hits. |
| libName | String | Human | The name of the custom text library that contains the custom term hit by the text entry. |

logoData

| Parameter | Type | Example | Description |
|---|---|---|---|
| type | String | TV | The type of the detected logo. For example, a value of TV indicates the logo of a TV station. |
| name | String | ***Station | The name of the detected logo. |
| x | Float | 140 | The distance between the upper-left corner of the logo area and the y-axis, with the upper-left corner of the image being the coordinate origin. Unit: pixel. |
| y | Float | 68 | The distance between the upper-left corner of the logo area and the x-axis, with the upper-left corner of the image being the coordinate origin. Unit: pixel. |
| w | Float | 106 | The width of the logo area. Unit: pixel. |
| h | Float | 106 | The height of the logo area. Unit: pixel. |

face

| Parameter | Type | Example | Description |
|---|---|---|---|
| name | String | xxxx | The name of the recognized face that is detected in the moderated image. |
| rate | Float | 97.03 | The probability that the detected face in the moderated image hits the recognized face. |

| Parameter | Type | Example | Description |
|---|---|---|---|
| id | String | AliFace_001**<br>** | The ID of the face that you want to delete. |

hitLibInfo

| Parameter | Type | Example | Description |
|---|---|---|---|
| context | String | xxxx | The custom text that the detected text hits. |
| libCode | String | 69751 | The code of the library that contains the custom text hit by the detected text. |
| libName | String | Human | The name of the library that contains the custom text hit by the detected text. |

hintWordsInfo

| Parameter | Type | Example | Description |
|---|---|---|---|
| context | String | xxxx | The term that the detected text hits. |

## Examples

### Sample requests

```
http(s)://[Endpoint]/green/video/results
&<Common request parameters>
[
    "taskId****",
    "taskId****"
]
```

### Sample success responses

- Detect only the images in a video

```
{
    "code": 200,
    "msg": "OK",
    "requestId": "requestID****",
    "data": [
        {
            "code": 200,
            "msg": "OK",
            "dataId": "videoId****",
            "taskId": "taskId****",
            "results": [
                {
                    "label": "porn",
                    "rate": 99.2,
                    "scene": "porn",
                    "suggestion": "block"
                }
            ]
        }
    ]
}
```

- Detect both the images and audio in a video

```
{
    "code": 200,
    "msg": "OK",
    "requestId": "requestID****",
    "data": [
        {
            "code": 200,
            "msg": "OK",
            "dataId": "videoId****",
            "taskId": "taskId****",
            "results": [
                {
                    "label": "porn",
                    "rate": 99.2,
                    "scene": "porn",
                    "suggestion": "block"
                }
            ],
            "audioScanResults": [
                {
                    "scene": "antispam",
                    "label": "customized",
                    "suggestion": "block",
                    "rate": 99.91,
                    "details": [
                        {
                            "startTime": 0,
                            "endTime": 24,
                            "text": "Computer",
                            "label": "customized"
                        },
                        {
                            "startTime": 24,
                            "endTime": 60,
                            "text": "Computer",
                            "label": "normal"
                        }
                    ]
                }
            ]
        }
    ]
}
```

## 5.2.2. Stop moderation tasks

This topic describes how to call an operation to stop video stream moderation tasks. It intends to help you construct an HTTP call request.

### Operation description

Operation: /green/video/cancelscan. After you submit asynchronous video stream moderation tasks, you can call this operation at any time to stop the video stream moderation tasks that are in progress. You can call this operation to stop moderating only video streams, but not videos.

You can call this operation to stop moderation tasks that are in progress.

Billing method: This operation is free of charge.

## Request parameters

For more information about the common request parameters that must be included in all Content Moderation API requests, see Common parameters.

The request body is a JSON structure. The following table describes the parameters that are contained in the request body.

| Parameter | Type | Required | Example | Description |
|---|---|---|---|---|
| body | JSONArray | Yes | ["taskId1", "taskId2"] | The list of IDs of moderation tasks you want to stop. You can specify up to 100 task IDs. |

## Response parameters

The data parameter in the response body is a JSON array. The following table describes the parameters that are contained in each element of the JSON array.

| Parameter | Type | Example | Description |
|---|---|---|---|
| code | Integer | 200 | |
| msg | String | OK | The message that is returned for the request. |
| dataId | String | dataId1 | The ID of the moderated video, which you specify in the dataId parameter of the moderation request. |
| taskId | String | taskId1 | The ID of the moderation task, which is returned by the Content Moderation server to uniquely identify the moderation task. |

## Examples

### Sample requests

```
[
  "taskId1",
  "taskId2"
]
```

### Sample success responses

```
{
    "code":200,
    "msg":"OK",
    "requestId":"AD87F1D1-A675-4F82-8D67-ED43B6FFA876",
    "data":[
        {
            "code":200,
            "msg":"ok",
            "dataId":"dataId1",
            "taskId":"taskId1"
        },
        {
            "code":200,
            "msg":"ok",
            "dataId":"dataId2",
            "taskId":"taskId2"
        }
    ]
}
```

# 5.2.3. Give feedback on moderation results

If you find that video moderation results returned by Alibaba Cloud Content Moderation are incorrect, you can call an operation to provide feedback. Alibaba Cloud Content Moderation will optimize the algorithm model based on your feedback.

## Description

Operation: /green/video/feedback

You can call this operation to give feedback on the result of a video moderation task and provide your expected result.

> **Note** You are not charged for calling this operation.

## Request parameters

For more information about the common request parameters that must be included in all Content Moderation API requests, see Common request parameters.

The request body is a JSON object. The following table describes the parameters that the JSON object contains.

| Parameter | Type | Required | Description |
|---|---|---|---|
| taskId | String | Yes | The ID returned by the Content Moderation server, which uniquely identifies the moderation task. |
| dataId | String | No | The ID of the moderated video, which you specify in the dataId parameter of the moderation request. |

| Parameter | Type | Required | Description |
|---|---|---|---|
| url | String | No | The URL of the moderated video, which you specify in the moderation request. If you have not specified the video URL in the moderation request, do not specify this parameter here. |
| suggestion | String | No | The moderation result you expect to return. Valid values:<br>- *pass*: The video is normal.<br>- *block*: The video contains violations. |
| scenes | String array | No | The scenario of video moderation. Valid values:<br>- *porn*: pornography detection<br>- *terrorism*: terrorist content detection<br>- *ad*: ad violation detection<br>**Note** You can specify multiple moderation scenarios. For example, you can specify both porn and terrorism in the `scenes` parameter to indicate pornography detection and terrorist content detection scenarios. |
| frames | JSON array | Yes | The information about captured frames that you consider to be in the detected category. Each element in the JSON array of the frames parameter is a structure. For more information about the structure of each element, see frame. |

frame

| Parameter | Type | Required | Description |
|---|---|---|---|
| url | String | Yes | The URL of the captured frame. |
| offset | Integer | Yes | The interval between the start of the video and the captured frame. Unit: seconds. |
| note | String | No | The description of the captured frame. |

## Response parameters

For more information about common response parameters that this operation returns, see Common response parameters.

The data parameter in the response body is empty.

## Examples

### Sample requests

```
{
  "taskId":"taskId xxx",
  "dataId":"dataId xxx",
  "url":"http://a.b.c/d.mp4",
  "frames":[
    {
      "url":"http://a.b.c/1.jpg",
      "offset":100
    },
    {
      "url":"http://a.b.c/2.jpg",
      "offset":200
    }
  ],
  "suggestion": "block",
  "scenes":[
    "ad",
    "terrorism"
  ],
  "note": "blabla"
}
```

### Sample success responses

```
{
  "code": 200,
  "msg": "OK",
  "requestId": "EE5A1189-4D7B-4C24-AD78-4C1FAA3E7A0C"
}
```

# 5.3. Text moderation

## 5.3.1. Text Synchronous moderation

This topic describes the /green/text/scan operation that you can call to moderate text. You can call this operation to moderate text for violations, such as pornographic content, ads, excessive junk content, political content, and abuse.

### Operation description

Operation: /green/text/scan

You can call this operation to submit text moderation tasks.

Billing method:

### Request parameters

| Parameter | Type | Required | Example | Description |
|---|---|---|---|---|
| bizType | String | No | default | |

| Parameter | Type | Required | Example | Description |
|---|---|---|---|---|
| scenes | String Array | Yes | ["antispam"] | The moderation scenario. Set the value to *antispam*, which indicates text moderation. |
| tasks | JSON Array | Yes | | The list of text moderation tasks. The JSON array contains one or more elements. Each element is a structure. You can add up to 100 elements to the array. In other words, you can submit up to 100 text entries at a time. For more information about the structure of each element, see task. |

task

| Parameter | Type | Required | Example | Description |
|---|---|---|---|---|
| clientInfo | JSON Object | No | | |
| dataId | String | No | cfd33235-71a4-468b-8137-a5ffe323a7e8 | |
| content | String | Yes | We are a micro-lending company, which provides an unsecured loan on the same day with secure, fast, convenient, and door-to-door services when you apply for a loan on your mobile phone. | The content of the text to be moderated. Each text entry can be up to 10,000 characters in length, including punctuation marks. |

## Response parameters

| Parameter | Type | Example | Description |
|---|---|---|---|
| code | Integer | 200 | |
| msg | String | OK | The message that is returned for the request. |
| dataId | String | cfd33235-71a4-468b-8137-a5ffe323a7e8 | |
| taskId | String | txt6HB8NQoEbU@5fosnj2xVEM-1tAokr | The ID of the moderation task. |
| content | String | We are a micro-lending company, which provides an unsecured loan on the same day with secure, fast, convenient, and door-to-door services when you apply for a loan on your mobile phone. | The content of the moderated text, which you specify in the moderation request. |

| Parameter | Type | Example | Description |
|---|---|---|---|
| filteredContent | String | We are a **** company, which provides an *** loan on the same day with secure, fast, convenient, and **** services when you apply for a loan on your mobile phone. | The text that is returned if the original text hits terms in the custom term library, with the terms being replaced with asterisks (*). |
| results | JSONArray | | The return results. If HTTP status code 200 is returned after a successful call, the array in the return results contains one or more elements. Each element is a structure. For more information about the structure, see result. |

result

| Parameter | Type | Example | Description |
|---|---|---|---|
| scene | String | antispam | The moderation scenario, which you specify in the moderation request. |
| suggestion | String | block | The recommended subsequent operation for you to perform. Valid values:<br>• *pass*: The text is normal.<br>• *review*: The text requires human review.<br>• *block*: The text contains violations and can be deleted or blocked. |

| Parameter | Type | Example | Description |
|---|---|---|---|
| label | String | porn | The category of the moderation result for the moderated text. Valid values:<br>- *normal*: normal<br>- *spam*: junk content<br>- *ad*: ad<br>- *politics*: political content<br>- *terrorism*: terrorist content<br>- *abuse*: abuse<br>- *porn*: pornographic content<br>- *flood*: excessive junk content<br>- *contraband*: prohibited content<br>- *meaningless*: meaningless content<br>- *customized*: custom content, such as a custom term |
| rate | Float | 99.91 | |
| extras | JSONObject | {"userId":"xxx"} | The additional information. This is an extended parameter. |
| details | JSONArray | | The details of the risky content that the moderated text hits. A text entry can hit multiple pieces of risky content. For more information about the structure, see detail. |

### detail

| Parameter | Type | Example | Description |
|---|---|---|---|
| label | String | porn | The category of the risky content that the moderated text hits. Valid values:<br>- *spam*: junk content<br>- *ad*: ad<br>- *politics*: political content<br>- *terrorism*: terrorist content<br>- *abuse*: abuse<br>- *porn*: pornographic content<br>- *flood*: excessive junk content<br>- *contraband*: prohibited content<br>- *meaningless*: meaningless content<br>- *customized*: custom content, such as a custom term |
| contexts | JSONArray | | The context information of the risky content that the moderated text hits. For more information about the structure, see context. |

| Parameter | Type | Example | Description |
|---|---|---|---|
| hintWords | JSONArray |  | The information about the term that the moderated text hits, which is used to indicate the reason for the violation. Multiple terms that the text hits may be returned. For more information about the structure, see hintWord.<br>ⓘ **Note** By default, this parameter is not returned. If you want this parameter to be returned, submit a . |

context

| Parameter | Type | Example | Description |
|---|---|---|---|
| context | String | door-to-door | The term that the moderated text hits. If the text hits a term, the term is returned. If the text hits the algorithmic model, this parameter is not returned. |
| positions | JSONArray | [{"startPos":1, "endPos":10}] | The position of the term that the moderated text hits in the original text. |
| libName | String | Name of your custom text library | The name of the custom text library. This parameter is returned if the moderated text hits a term in the custom text library. |
| libCode | String | 123456 | The code of the custom text library. This parameter is returned if the moderated text hits a term in the custom text library. |
| ruleType | String | ip | The behavior rule. This parameter is returned if the moderated text hits the behavior rule. Valid values:<br>• *user_id*<br>• *ip*<br>• *umid*<br>• *content*<br>• *similar_content*<br>• *imei*<br>• *imsi* |

hintWord

| Parameter | Type | Example | Description |
|---|---|---|---|
| context | String | door-to-door | The term that the moderated text hits. |

## Examples

Sample requests

```
{
     "scenes": [
        "antispam"
    ],
     "tasks": [
          {
               "dataId": "cfd33235-71a4-468b-8137-a5ffe323a7e8",
               "content": "We are a micro-lending company, which provides an unsecured loan
on the same day with secure, fast, convenient, and door-to-door services when you apply for
a loan on your mobile phone."
        }
    ]
}
```

Sample success responses

```
{
  "code": 200,
  "data": [
    {
      "code": 200,
      "content": "We are a micro-lending company, which provides an unsecured loan on the s
ame day with secure, fast, convenient, and door-to-door services when you apply for a loan
on your mobile phone.",
      "dataId": "cfd33235-71a4-468b-8137-a5ffe323a7e8",
      "filteredContent": "We are a **** company, which provides an *** loan on the same day
with secure, fast, convenient, and **** services when you apply for a loan on your mobile p
hone.",
      "msg": "OK",
      "results": [
        {
          "details": [
            {
              "contexts": [
                {
                  "context": "unsecured",
                  "positions": [
                    {
                      "endPos": 19,
                      "startPos": 16
                    }
                  ]
                },
                {
                  "context": "micro-lending",
                  "positions": [
                    {
                      "endPos": 6,
                      "startPos": 2
                    }
                  ]
                }
              ],
              "label": "spam"
            },
            {
              "contexts": [
                {
                  "context": "door-to-door",
                  "libCode": "123456",
                  "libName": "Name of your custom text library",
                  "positions": [
                    {
                      "endPos": 34,
                      "startPos": 30
                    }
                  ]
                }
              ],
              "label": "porn"
            }
          ],
          "label": "porn",
          "rate": 99.91,
          "scene": "antispam",
          "suggestion": "block"
        }
      ],
      "taskId": "txt6HB8NQoEbU@5fosnj2xVEM-1tAokr"
    }
  ],
  "msg": "OK",
  "requestId": "25711794-BF6D-4F32-A735-09CA21197D32"
}
```

## 5.3.2. Give feedback on moderation results

If you find that text moderation results returned by Alibaba Cloud Content Moderation are incorrect, you can call an operation to provide feedback.

### Description

**Operation**: /green/text/feedback. You can call this operation to give feedback on the result of a text moderation task.

### Request parameters

For more information about the common request parameters that must be included in all Content Moderation API requests, see Common request parameters.

The request body is a JSON array. The following table describes the parameters that each element of the JSON array contains.

| Parameter | Type | Required | Description |
|---|---|---|---|
| taskId | String | Yes | The ID returned by the Content Moderation server, which uniquely identifies the moderation task. |
| dataId | String | No | The ID of the moderated text, which you specify in the dataId parameter of the moderation request. |
| content | String | No | The content of the moderated text. Each piece of text can be up to 10,000 characters in length. |
| label | String | No | The expected category of moderation results for the moderated text in the specified moderation scenario. For more information about valid values, see Parameters scenes and label of the /green/text/scan operation.<br>Note If you set the label parameter to normal, the system adds the text to the whitelist of the feedback-based text library in the Alibaba Cloud Content Moderation console. If you set the label parameter to a value other than normal, the system adds the text to the blacklist of the feedback-based text library. |
| note | String | No | The description of the moderated text, such as the keywords in the text. |

## Response parameters

For more information about common response parameters that this operation returns, see Common response parameters.

The data parameter in the response body is empty.

## Examples

### Sample requests

```
{
  "dataId": "test7fcmzGkKbNp7@JqPQRnwIe-1mwwZX",
  "taskId": "taskid xxx",
  "content": "foobar",
  "label": "spam",
  "note": "blabla ..."
}
```

### Sample success responses

```
{
  "msg": "OK",
  "code": 200,
  "requestId": "02C418E7-81D4-4375-9732-900B64D07CEE"
}
```

# 6.Image OCR moderation

## 6.1. Image OCR

### 6.1.1. Detect text in images synchronously

This topic describes the /green/image/scan operation that you can call to submit optical character recognition (OCR) tasks and obtain OCR results in real time. You can submit the OCR tasks to detect and obtain text in images.

#### Description

Operation: /green/image/scan

You can call this operation to submit synchronous OCR tasks.

- **Billing method:**
- **Response time:**
- **Return results:**
- **Limits on images:**

#### Request parameters

| Parameter | Type | Required | Description |
|---|---|---|---|
| bizType | String | No | |
| scenes | String array | Yes | The moderation scenario. Set the value to *ocr*. |
| tasks | JSON array | Yes | The list of OCR tasks. Each element in the JSON array is an OCR task structure, namely, each element corresponds to an image. The JSON array can contain a maximum of 10 elements, namely, you can submit a maximum of 10 images at a time. For more information about the structure of each element, see task. |

task

| Parameter | Type | Required | Description |
|---|---|---|---|
| dataId | String | No | The ID of the image to be moderated. Make sure that each ID is unique in a request. |
| url | String | Yes | The URL of the image to be moderated. |
| interval | Integer | No | |
| maxFrames | Integer | No | |

#### Response parameters

| Parameter | Type | Description |
|---|---|---|
| code | Integer | The HTTP status code returned for the OCR task. |
| msg | String | The message returned for the OCR task. |
| dataId | String | |
| taskId | String | The ID of the OCR task. |
| url | String | The URL of the moderated image. |
| results | Array | The return results of the OCR task. If HTTP status code 200 is returned after a successful call, the array in the return results contains one or more elements. Each element is a structure. For more information about the structure of each element, see result. |

result

| Parameter | Type | Description |
|---|---|---|
| scene | String | The moderation scenario. The value is fixed to *ocr*. |
| label | String | The category of the OCR results. Valid values:<br>● *normal*: The image does not contain text.<br>● *ocr*: The image contains text. |
| suggestion | String | The machine-assisted moderation result of the moderated image. Valid values:<br>● *pass*: The image does not require further actions.<br>● *review*: The image requires human review. |
| rate | Floating point | The probability that the moderated image falls into the detected category. You can ignore this parameter in the OCR scenario. |
| ocrLocations | Array | The information about the single text entry in the moderated static image, which includes the text, text size, and text location. For more information about the structure, see ocrLocation. |
| ocrData | Array | The combination of all text in the moderated static image. In general, the text combination is stored as the first element of the array. |
| frames | Array | The frames that are captured from the moderated GIF image and the text that is detected in each frame. |

## Examples

Sample requests

```
{
    "scenes": [
        "ocr"
    ],
    "tasks": [
        {
            "dataId": "test_data_xxxx",
            "url": "https://test_image_xxxx.png"
        }
    ]
}
```

Sample success responses

```
{
    "code": 200,
    "data": [
        {
            "code": 200,
            "dataId": "test_data_xxxx",
            "extras": {
            },
            "msg": "OK",
            "results": [
                {
                    "label": "ocr",
                    "ocrData": [
                        "hello, this is a test text."
                    ],
                    "ocrLocations": [
                        {
                            "h": 26,
                            "text": "hello",
                            "w": 83,
                            "x": 41,
                            "y": 84
                        },
                        {
                            "h": 25,
                            "text": " this is a test text.",
                            "w": 95,
                            "x": 78,
                            "y": 114
                        }
                    ],
                    "rate": 99.91,
                    "scene": "ocr",
                    "suggestion": "review"
                }
            ],
            "taskId": "img5A@k7a@B4q@6K@d9nfKgOs-1sWeLu",
            "url": "https://test_image_xxxx.png"
        }
    ],
    "msg": "OK",
    "requestId": "C4AB08A9-AD75-4410-859B-0B9EF6DFC3C4"
}
```

## 6.1.2. Detect text in images asynchronously

This topic describes the /green/image/asyncscan operation that you can call to submit asynchronous optical character recognition (OCR) tasks. You can submit the OCR tasks to detect and obtain text in images.

### Submit asynchronous OCR tasks

Operation: /green/image/asyncscan

You can call this operation to submit asynchronous OCR tasks.

- **Billing method:**
- **Response time:**
- **Return results:**
  - Enable callback notification to obtain OCR results: When you submit asynchronous OCR tasks, you can specify a callback URL for receiving OCR results in the callback parameter of the OCR request. For more information about the callback parameter, see Request parameters.
  - Poll OCR results at regular intervals: You do not need to specify the callback parameter when you submit asynchronous OCR tasks. After you submit the tasks, you can call the /green/image/results operation to query OCR results. For more information about the /green/image/results operation, see Query asynchronous OCR results.
- **Limits on images:**

## Request parameters

| Parameter | Type | Required | Description |
|---|---|---|---|
| bizType | String | No | |
| scenes | String array | Yes | The moderation scenario. Set the value to *ocr*. |
| callback | String | No | |
| seed | String | No | |
| tasks | JSON array | Yes | The list of OCR tasks. Each element in the JSON array is an OCR task structure and corresponds to an image. The JSON array can contain a maximum of 100 elements. In other words, you can submit a maximum of 100 images at a time. For more information about the structure of each element, see task. |

task

| Parameter | Type | Required | Description |
|---|---|---|---|
| dataId | String | No | The ID of the image to be moderated. Make sure that each ID is unique in a request. |
| url | String | Yes | The URL of the image to be moderated. |
| interval | Integer | No | |
| maxFrames | Integer | No | |

## Response parameters

| Parameter | Type | Description |
|---|---|---|
| code | Integer | The HTTP status code that is returned for the OCR task. |
| msg | String | The message that is returned for the OCR task. |
| dataId | String | |
| taskId | String | The ID of the OCR task. |
| url | String | The URL of the moderated image. |
| extras | JSON structure | |

## Examples

### Sample requests

```
{
    "scenes": [
        "ocr"
    ],
    "tasks": [
        {
            "dataId": "test_data_xxxx",
            "url": "https://test_image_xxxx.png"
        }
    ]
}
```

### Sample success responses

```
{
    "code": 200,
    "msg": "OK",
    "requestId": "92AD868A-F5D2-4AEA-96D4-E1273B8E074C",
    "data": [
        {
            "code": 200,
            "msg": "OK",
            "dataId": "test_data_xxxx",
            "taskId": "aaa25f95-4892-4d6b-aca9-7939bc6e9baa-1486198766695",
            "url": "https://test_image_xxxx.png"
        }
    ]
}
```

## Query asynchronous OCR results

Operation: /green/image/results

You can call this operation to query asynchronous OCR results.

- **Billing method:**
- **Response time:**

## Request parameters

| Parameter | Type | Required | Description |
|---|---|---|---|
| body | JSON array | Yes | |

## Response parameters

| Parameter | Type | Description |
|---|---|---|
| code | Integer | The HTTP status code that is returned for the OCR task. |
| msg | String | The message that is returned for the OCR task. |
| dataId | String | |
| taskId | String | The ID of the OCR task. |
| url | String | The URL of the moderated image. |
| results | Array | The return results of the OCR task. If HTTP status code 200 is returned after a successful call, the array in the return results contains one or more elements. Each element is a structure. For more information about the structure of each element, see result. |

result

| Parameter | Type | Description |
|---|---|---|
| scene | String | The moderation scenario. The value is fixed to *ocr*. |
| label | String | The category of the OCR results. Valid values:<br>• *normal*: The image does not contain text.<br>• *ocr*: The image contains text. |
| suggestion | String | The machine-assisted moderation result of the moderated image. Valid values:<br>• *pass*: The image does not require further actions.<br>• *review*: The image requires human review. |
| rate | Floating point | The probability that the moderated image falls into the detected category. You can ignore this parameter in the OCR scenario. |
| ocrLocations | Array | The information about the single text entry in the moderated static image, which includes the text, text size, and text location. For more information about the structure, see ocrLocation. |

| Parameter | Type | Description |
|---|---|---|
| ocrData | Array | The combination of all text in the moderated static image. In general, the text combination is stored as the first element of the array. |
| frames | Array | The frames that are captured from the moderated GIF image and the text that is detected in each frame. |

## Examples

Sample requests

```
[
    "aaa25f95-4892-4d6b-aca9-7939bc6e9baa-1486198766695"
]
```

Sample success responses

```
{
    "code": 200,
    "data": [
        {
            "code": 200,
            "dataId": "test_data_xxxx",
            "extras": {
            },
            "msg": "OK",
            "results": [
                {
                    "label": "ocr",
                    "ocrData": [
                        "This topic describes how to call an operation to submit asynchronous image moderation tasks."
                    ],
                    "ocrLocations": [
                        {
                            "h": 19,
                            "text": "This topic describes how to call an operation to submit asynchronous image moderation tasks.",
                            "w": 362,
                            "x": 31,
                            "y": 11
                        }
                    ],
                    "rate": 99.91,
                    "scene": "ocr",
                    "suggestion": "review"
                }
            ],
            "taskId": "aaa25f95-4892-4d6b-aca9-7939bc6e9baa-1486198766695",
            "url": "https://test_image_xxxx.png"
        }
    ],
    "msg": "OK",
    "requestId": "992C7849-AA45-4055-8F82-8D44D64C15E3"
}
```