智能编辑

API 文档

产品文档

腾讯云

文档目录

API 文档
更新历史

最近更新时间：2022-07-05 06:14:28

第 12 次发布

发布时间：2022-07-05 06:14:24

本次发布包含了以下内容：

改善已有的文档。

新增数据结构：

- DynamicImageInfo
- MediaCuttingWatermark
- MediaCuttingWatermarkImage
- MediaCuttingWatermarkText
- SpriteImageInfo

修改数据结构：

- MediaCuttingInfo
  - 新增成员：WatermarkInfoSet, DropPureColor
- MediaCuttingOutForm
  - 新增成员：SpriteInfo, DynamicInfo
- MediaCuttingTaskResult
  - 新增成员：ImageCount
- MediaJoiningInfo
  - 新增成员：Mode
- SaveInfo
  - 新增成员：Id
- TaskResultFile
  - 新增成员：Md5

第 11 次发布

发布时间：2021-07-23 08:01:48

本次发布包含了以下内容：

改善已有的文档。

新增数据结构：

- TextMarkInfoItem

修改数据结构：

- VideoInfo
  - 新增成员：TextMarkInfo

第 10 次发布

发布时间：2021-06-09 08:01:32

本次发布包含了以下内容：

改善已有的文档。

新增数据结构：

- VideoRepair
- VideoSuperResolution

修改数据结构：

- VideoEnhance
  - 新增成员：VideoSuperResolution, VideoRepair

第 9 次发布

发布时间：2021-05-13 08:01:13

本次发布包含了以下内容：

改善已有的文档。

新增数据结构：

- AudioEnhance
- LoudnessInfo
- RemoveReverb

修改数据结构：

- AudioInfo
  - 新增成员：LoudnessInfo, AudioEnhance, RemoveReverb

第 8 次发布

发布时间：2021-04-12 08:01:01

本次发布包含了以下内容：

改善已有的文档。

新增数据结构：

- MediaResultInfo
- ResultAudioInfo
- ResultVideoInfo

修改数据结构：

- TaskResultFile
  - 新增成员：FileSize, MediaInfo

第 7 次发布

发布时间：2021-04-01 08:00:55

本次发布包含了以下内容：

改善已有的文档。

修改数据结构：

- AudioInfo
  - 新增成员：EnableMuteAudio
- MuxInfo
  - 新增成员：FlvFlags

第 6 次发布

发布时间：2021-03-01 08:00:57

本次发布包含了以下内容：

改善已有的文档。

新增数据结构：

- HiddenMarkInfo

修改数据结构：

- VideoInfo
  - 新增成员：HiddenMarkInfo

第 5 次发布

发布时间：2021-01-07 08:00:44

本次发布包含了以下内容：

改善已有的文档。

新增数据结构：

- FrameTagItem
- FrameTagRec
- FrameTagResult
- MediaRecognitionInfo
- MediaRecognitionTaskResult
- SubtitleItem
- SubtitleRec
- SubtitleResult
- TagItem

修改数据结构：

- MediaProcessInfo
  - 新增成员：MediaRecognitionInfo
- MediaProcessTaskResult
  - 新增成员：MediaRecognitionTaskResult

第 4 次发布

发布时间：2020-12-24 08:00:54

本次发布包含了以下内容：

改善已有的文档。

新增数据结构：

- Denoise

修改数据结构：

- AudioInfo
  - 新增成员：Denoise

第 3 次发布

发布时间：2020-12-21 08:01:10

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- CreateMediaProcessTask
- DescribeMediaProcessTaskResult
- StopMediaProcessTask

新增数据结构：

- IntervalTime
- MediaCuttingInfo
- MediaCuttingOutForm
- MediaCuttingTaskResult
- MediaCuttingTimeInfo
- MediaJoiningInfo
- MediaJoiningTaskResult
- MediaProcessInfo
- MediaProcessTaskResult
- MediaSourceInfo
- MediaTargetInfo
- SectionTime
- TargetVideoInfo
- TaskResultFile

第 2 次发布

发布时间：2020-09-17 08:00:24

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- CreateMediaQualityRestorationTask
- CreateQualityControlTask
- DescribeMediaQualityRestorationTaskRusult
- DescribeQualityControlTaskResult
- StopMediaQualityRestorationTask

新增数据结构：

- ArtifactReduction

- AudioInfo
- AudioInfoResultItem
- ColorEnhance
- DarInfo
- Denoising
- EditInfo
- FaceProtect
- FileInfo
- LowLightEnhance
- MediaQualityRestorationTaskResult
- MuxInfo
- PicMarkInfoItem
- QualityControlInfo
- QualityControlInfoTaskResult
- QualityControlItem
- QualityControlResultItems
- ScratchRepair
- SegmentInfo
- Sharp
- SubTaskResultItem
- SubTaskTranscodeInfo
- TargetInfo
- VideoEnhance
- VideoInfo
- VideoInfoResultItem

第 1 次发布

发布时间：2020-03-23 16:46:34

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- CreateEditingTask
- DescribeEditingTaskResult

新增数据结构：

- CallbackInfo
- ClassificationEditingInfo
- ClassificationTaskResult
- ClassificationTaskResultItem
- CosAuthMode
- CosInfo
- CoverEditingInfo
- CoverTaskResult
- CoverTaskResultItem
- DownInfo
- EditingInfo
- EditingTaskResult

- HighlightsEditingInfo
- HighlightsTaskResult
- HighlightsTaskResultItem
- HighlightsTaskResultItemSegment
- OpeningEndingEditingInfo
- OpeningEndingTaskResult
- OpeningEndingTaskResultItem
- SaveInfo
- StripEditingInfo
- StripTaskResult
- StripTaskResultItem
- TagEditingInfo
- TagTaskResult
- TagTaskResultItem
- UrlInfo

简介

最近更新时间：2021-06-23 11:28:47

欢迎使用 智能编辑 API 3.0 版本。全新的 API 接口文档更加规范和全面，统一的参数风格和公共错误码，统一的 SDK/CLI 版本与 API 文档严格一致，给您带来简单快捷的使用体验。支持全地域就近接入让您更快连接腾讯云产品。更多腾讯云 API 3.0 使用介绍请查看：快速入门

智能编辑（Intelligent Editing，IE）利用 AI 以及腾讯多年的音视频技术，多维度、全方位理解视频内容，为用户提供视频分类、标签、封面、精彩集锦、拆条等多种基础能力，辅助视频内容分析和生产，让"采、编、存、发”内容生产流程更便捷。

API 概览

最近更新时间：2020-12-21 08:01:15

画质重生任务相关接口

接口名称	接口功能
CreateMediaQualityRestorationTask	创建画质重生任务
DescribeMediaQualityRestorationTaskRusult	获取画质重生任务结果
StopMediaQualityRestorationTask	删除画质重生任务

媒体质检任务相关接口

接口名称	接口功能
CreateQualityControlTask	创建媒体质检任务
DescribeQualityControlTaskResult	获取媒体质检任务结果

编辑理解相关接口

接口名称	接口功能
CreateEditingTask	创建编辑理解任务
DescribeEditingTaskResult	获取编辑理解任务结果

编辑处理相关接口

接口名称	接口功能
CreateMediaProcessTask	创建编辑处理任务
DescribeMediaProcessTaskResult	获取编辑处理任务结果
StopMediaProcessTask	停止编辑处理任务

调用方式
请求结构

最近更新时间：2020-09-28 08:00:46

1. 服务地址

API 支持就近地域接入，本产品就近地域接入域名为 ie.tencentcloudapi.com ，也支持指定地域域名访问，例如广州地域的域名为 ie.ap-guangzhou.tencentcloudapi.com 。

推荐使用就近地域接入域名。根据调用接口时客户端所在位置，会自动解析到**最近的**某个具体地域的服务器。例如在广州发起请求，会自动解析到广州的服务器，效果和指定 ie.ap-guangzhou.tencentcloudapi.com 是一致的。

注意：对时延敏感的业务，建议指定带地域的域名。

注意：域名是 API 的接入点，并不代表产品或者接口实际提供服务的地域。产品支持的地域列表请在调用方式/公共参数文档中查阅，接口支持的地域请在接口文档输入参数中查阅。

目前支持的域名列表为：

接入地域	域名
就近地域接入（推荐，只支持非金融区）	ie.tencentcloudapi.com
华南地区(广州)	ie.ap-guangzhou.tencentcloudapi.com
华东地区(上海)	ie.ap-shanghai.tencentcloudapi.com
华北地区(北京)	ie.ap-beijing.tencentcloudapi.com
西南地区(成都)	ie.ap-chengdu.tencentcloudapi.com
西南地区(重庆)	ie.ap-chongqing.tencentcloudapi.com
港澳台地区(中国香港)	ie.ap-hongkong.tencentcloudapi.com
亚太东南(新加坡)	ie.ap-singapore.tencentcloudapi.com
亚太东南(曼谷)	ie.ap-bangkok.tencentcloudapi.com
亚太南部(孟买)	ie.ap-mumbai.tencentcloudapi.com
亚太东北(首尔)	ie.ap-seoul.tencentcloudapi.com
亚太东北(东京)	ie.ap-tokyo.tencentcloudapi.com
美国东部(弗吉尼亚)	ie.na-ashburn.tencentcloudapi.com
美国西部(硅谷)	ie.na-siliconvalley.tencentcloudapi.com
北美地区(多伦多)	ie.na-toronto.tencentcloudapi.com
欧洲地区(法兰克福)	ie.eu-frankfurt.tencentcloudapi.com
欧洲地区(莫斯科)	ie.eu-moscow.tencentcloudapi.com

2. 通信协议

腾讯云 API 的所有接口均通过 HTTPS 进行通信，提供高安全性的通信通道。

3. 请求方法

支持的 HTTP 请求方法:

- POST（推荐）
- GET

POST 请求支持的 Content-Type 类型:

- application/json（推荐），必须使用签名方法 v3（TC3-HMAC-SHA256）。
- application/x-www-form-urlencoded，必须使用签名方法 v1（HmacSHA1 或 HmacSHA256）。
- multipart/form-data（仅部分接口支持），必须使用签名方法 v3（TC3-HMAC-SHA256）。

GET 请求的请求包大小不得超过32KB。POST 请求使用签名方法 v1（HmacSHA1、HmacSHA256）时不得超过1MB。POST 请求使用签名方法 v3（TC3-HMAC-SHA256）时支持10MB。

4. 字符编码

均使用 UTF-8 编码。

公共参数

最近更新时间：2022-04-12 06:15:03

公共参数是用于标识用户和接口签名的参数，如非必要，在每个接口单独的接口文档中不再对这些参数进行说明，但每次请求均需要携带这些参数，才能正常发起请求。

公共参数的具体内容会因您使用的签名方法版本不同而有所差异。

使用签名方法 v3 的公共参数

签名方法 v3（有时也称作 TC3-HMAC-SHA256）相比签名方法 v1（有些文档可能会简称签名方法），更安全，支持更大的请求包，支持 POST JSON 格式，性能有一定提升，推荐使用该签名方法计算签名。完整介绍详见 签名方法 v3。

注意：接口文档中的示例由于目的是展示接口参数用法，简化起见，使用的是签名方法 v1 GET 请求，如果依旧想使用签名方法 v1 请参考下文章节。

使用签名方法 v3 时，公共参数需要统一放到 HTTP Header 请求头部中，如下表所示：

参数名称	类型	必选	描述
Action	String	是	HTTP 请求头：X-TC-Action。操作的接口名称。取值参考接口文档中输入参数公共参数 Action 的说明。例如云服务器的查询实例列表接口，取值为 DescribeInstances。
Region	String	-	HTTP 请求头：X-TC-Region。地域参数，用来标识希望操作哪个地域的数据。接口接受的地域取值参考接口文档中输入参数公共参数 Region 的说明。**注意：某些接口不需要传递该参数，接口文档中会对此特别说明，此时即使传递该参数也不会生效。**
Timestamp	Integer	是	HTTP 请求头：X-TC-Timestamp。当前 UNIX 时间戳，可记录发起 API 请求的时间。例如 1529223702。**注意：如果与服务器时间相差超过5分钟，会引起签名过期错误。**
Version	String	是	HTTP 请求头：X-TC-Version。操作的 API 的版本。取值参考接口文档中入参公共参数 Version 的说明。例如云服务器的版本 2017-03-12。
Authorization	String	是	HTTP 标准身份认证头部字段，例如： TC3-HMAC-SHA256 Credential=AKIDEXAMPLE/Date/service/tc3_request, SignedHeaders=content-type;host, Signature=fe5f80f77d5fa3beca038a248ff027d0445342fe2855ddc963176630326f1024 其中， - TC3-HMAC-SHA256：签名方法，目前固定取该值； - Credential：签名凭证，AKIDEXAMPLE 是 SecretId；Date 是 UTC 标准时间的日期，取值需要和公共参数 X-TC-Timestamp 换算的 UTC 标准时间日期一致；service 为产品名，通常为域名前缀，例如域名 cvm.tencentcloudapi.com 意味着产品名是 cvm。本产品取值为 ie； - SignedHeaders：参与签名计算的头部信息，content-type 和 host 为必选头部； - Signature：签名摘要，计算过程详见 文档。
Token	String	否	HTTP 请求头：X-TC-Token。即 安全凭证服务 所颁发的临时安全凭证中的 Token，使用时需要将 SecretId 和 SecretKey 的值替换为临时安全凭证中的 TmpSecretId 和 TmpSecretKey。使用长期密钥时不能设置此 Token 字段。
Language	String	否	HTTP 请求头：X-TC-Language。指定接口返回的语言，仅部分接口支持此参数。取值：zh-CN，en-US。zh-CN 返回中文，en-US 返回英文。

假设用户想要查询广州地域的云服务器实例列表中的前十个，接口参数设置为偏移量 Offset=0，返回数量 Limit=10，则其请求结构按照请求 URL、请求头部、请求体示例如下：

HTTP GET 请求结构示例：

```
https://cvm.tencentcloudapi.com/?Limit=10&Offset=0
```

```
Authorization: TC3-HMAC-SHA256 Credential=AKID********EXAMPLE/2018-10-09/cvm/tc3_request, SignedHeaders=content-type;host, Signature=5da7a33f6993f0614b047e5df4582db9e9bf4672ba50567dba16c6ccf174c474
Content-Type: application/x-www-form-urlencoded
Host: cvm.tencentcloudapi.com
X-TC-Action: DescribeInstances
X-TC-Version: 2017-03-12
X-TC-Timestamp: 1539084154
X-TC-Region: ap-guangzhou
```

HTTP POST（application/json）请求结构示例：

```
https://cvm.tencentcloudapi.com/

Authorization: TC3-HMAC-SHA256 Credential=AKID********EXAMPLE/2018-05-30/cvm/tc3_request, SignedHeaders=content-type;host, Signature=582c400e06b5924a6f2b5d7d672d79c15b13162d9279b0855cfba6789a8edb4c
Content-Type: application/json
Host: cvm.tencentcloudapi.com
X-TC-Action: DescribeInstances
X-TC-Version: 2017-03-12
X-TC-Timestamp: 1527672334
X-TC-Region: ap-guangzhou

{"Offset":0,"Limit":10}
```

HTTP POST（multipart/form-data）请求结构示例（仅特定的接口支持）：

```
https://cvm.tencentcloudapi.com/

Authorization: TC3-HMAC-SHA256 Credential=AKID********EXAMPLE/2018-05-30/cvm/tc3_request, SignedHeaders=content-type;host, Signature=582c400e06b5924a6f2b5d7d672d79c15b13162d9279b0855cfba6789a8edb4c
Content-Type: multipart/form-data; boundary=58731222010402
Host: cvm.tencentcloudapi.com
X-TC-Action: DescribeInstances
X-TC-Version: 2017-03-12
X-TC-Timestamp: 1527672334
X-TC-Region: ap-guangzhou

--58731222010402
Content-Disposition: form-data; name="Offset"

0
--58731222010402
Content-Disposition: form-data; name="Limit"

10
--58731222010402--
```

使用签名方法 v1 的公共参数

使用签名方法 v1（有时会称作 HmacSHA256 和 HmacSHA1），公共参数需要统一放到请求串中，完整介绍详见文档

参数名称	类型	必选	描述
Action	String	是	操作的接口名称。取值参考接口文档中输入参数公共参数 Action 的说明。例如云服务器的查询实例列表接口，取值为 DescribeInstances。
Region	String	-	地域参数，用来标识希望操作哪个地域的数据。接口接受的地域取值参考接口文档中输入参数公共参数 Region 的说明。**注意：某些接口不需要传递该参数，接口文档中会对此特别说明，此时即使传递该参数也不会生效。**
Timestamp	Integer	是	当前 UNIX 时间戳，可记录发起 API 请求的时间。例如1529223702，如果与当前时间相差过大，会引起签名过期错误。
Nonce	Integer	是	随机正整数，与 Timestamp 联合起来，用于防止重放攻击。
SecretId	String	是	在 云API密钥 上申请的标识身份的 SecretId，一个 SecretId 对应唯一的 SecretKey ，而 SecretKey 会用来生成请求签名 Signature。
Signature	String	是	请求签名，用来验证此次请求的合法性，需要用户根据实际的输入参数计算得出。具体计算方法参见文档。
Version	String	是	操作的 API 的版本。取值参考接口文档中入参公共参数 Version 的说明。例如云服务器的版本 2017-03-12。
SignatureMethod	String	否	签名方式，目前支持 HmacSHA256 和 HmacSHA1。只有指定此参数为 HmacSHA256 时，才使用 HmacSHA256 算法验证签名，其他情况均使用 HmacSHA1 验证签名。
Token	String	否	即 安全凭证服务 所颁发的临时安全凭证中的 Token，使用时需要将 SecretId 和 SecretKey 的值替换为临时安全凭证中的 TmpSecretId 和 TmpSecretKey。使用长期密钥时不能设置此 Token 字段。
Language	String	否	指定接口返回的语言，仅部分接口支持此参数。取值：zh-CN，en-US。zh-CN 返回中文，en-US 返回英文。

假设用户想要查询广州地域的云服务器实例列表，其请求结构按照请求 URL、请求头部、请求体示例如下：

HTTP GET 请求结构示例：

```
https://cvm.tencentcloudapi.com/?Action=DescribeInstances&Version=2017-03-12&SignatureMethod=HmacSHA256&Timestamp=1527672334&Signature=37ac2f4fde00b0ac9bd9eadeb459b1bbee224158d66e7ae5fcadb70b2d181d02&Region=ap-guangzhou&Nonce=23823223&SecretId=AKID********EXAMPLE

Host: cvm.tencentcloudapi.com
Content-Type: application/x-www-form-urlencoded
```

HTTP POST 请求结构示例：

```
https://cvm.tencentcloudapi.com/

Host: cvm.tencentcloudapi.com
Content-Type: application/x-www-form-urlencoded

Action=DescribeInstances&Version=2017-03-12&SignatureMethod=HmacSHA256&Timestamp=1527672334&Signature=37ac2f4fde00b0ac9bd9eadeb459b1bbee224158d66e7ae5fcadb70b2d181d02&Region=ap-guangzhou&Nonce=23823223&SecretId=AKID********EXAMPLE
```

地域列表

本产品所有接口 Region 字段的可选值如下表所示。如果接口不支持该表中的所有地域，则会在接口文档中单独说明。

地域	取值
华南地区(广州)	ap-guangzhou
华东地区(上海)	ap-shanghai
亚太东南(新加坡)	ap-singapore

签名方法 v3

最近更新时间：2022-04-18 06:13:29

以下文档说明了签名方法 v3 的签名过程，但仅在您编写自己的代码来调用腾讯云 API 时才有用。我们推荐您使用 腾讯云 API Explorer ，腾讯云 SDK 和 腾讯云命令行工具（TCCLI）等开发者工具，从而无需学习如何对 API 请求进行签名。

推荐使用 API Explorer　　点击调试

您可以通过 API Explorer 的【签名串生成】模块查看每个接口签名的生成过程。

腾讯云 API 会对每个请求进行身份验证，用户需要使用安全凭证，经过特定的步骤对请求进行签名（Signature），每个请求都需要在公共参数中指定该签名结果并以指定的方式和格式发送请求。

为什么要进行签名

签名通过以下方式帮助保护请求：

1. 验证请求者的身份
   签名确保请求是由持有有效访问密钥的人发送的。请参阅控制台 云 API 密钥 页面获取密钥相关信息。

2. 保护传输中的数据
   为了防止请求在传输过程中被篡改，腾讯云 API 会使用请求参数来计算请求的哈希值，并将生成的哈希值加密后作为请求的一部分，发送到腾讯云 API 服务器。服务器会使用收到的请求参数以同样的过程计算哈希值，并验证请求中的哈希值。如果请求被篡改，将导致哈希值不一致，腾讯云 API 将拒绝本次请求。

签名方法 v3（TC3-HMAC-SHA256）功能上覆盖了以前的签名方法 v1，而且更安全，支持更大的请求，支持 JSON 格式，POST 请求支持传空数组和空字符串，性能有一定提升，推荐使用该签名方法计算签名。

首次接触，建议使用 API Explorer 中的“签名串生成”功能，选择签名版本为“API 3.0 签名 v3”，可以生成签名过程进行验证，也可直接生成 SDK 代码。推荐使用腾讯云 API 配套的 7 种常见的编程语言 SDK，已经封装了签名和请求过程，均已开源，支持 Python、Java、PHP、Go、NodeJS、.NET、C++。

申请安全凭证

本文使用的安全凭证为密钥，密钥包括 SecretId 和 SecretKey。每个用户最多可以拥有两对密钥。

- SecretId：用于标识 API 调用者身份，可以简单类比为用户名。
- SecretKey：用于验证 API 调用者的身份，可以简单类比为密码。
- **用户必须严格保管安全凭证，避免泄露，否则将危及财产安全。如已泄漏，请立刻禁用该安全凭证。**

申请安全凭证的具体步骤如下：

1. 登录 腾讯云管理中心控制台 。
2. 前往 云API密钥 的控制台页面。
3. 在 云API密钥 页面，单击【新建密钥】即可以创建一对密钥。

签名版本 v3 签名过程

云 API 支持 GET 和 POST 请求。对于GET方法，只支持 Content-Type: application/x-www-form-urlencoded 协议格式。对于POST方法，目前支持 Content-Type: application/json 以及 Content-Type: multipart/form-data 两种协议格式，json 格式绝大多数接口均支持，multipart 格式只有特定接口支持，此时该接口不能使用 json 格式调用，参考具体业务接口文档说明。推荐使用 POST 请求，因为两者的结果并无差异，但 GET 请求只支持 32 KB 以内的请求包。

下面以云服务器查询广州区实例列表作为例子，分步骤介绍签名的计算过程。我们选择该接口是因为：

1. 云服务器默认已开通，该接口很常用；
2. 该接口是只读的，不会改变现有资源的状态；
3. 接口覆盖的参数种类较全，可以演示包含数据结构的数组如何使用。

在示例中，不论公共参数或者接口的参数，我们尽量选择容易犯错的情况。在实际调用接口时，请根据实际情况来，每个接口的参数并不相同，不要照抄这个例子的参数和值。此外，这里只展示了部分公共参数和接口输入参数，用户可以根据实际需要添加其他参数，例如 Language 和 Token 公共参数（在 HTTP 头部设置，添加 X-TC- 前缀）。

假设用户的 SecretId 和 SecretKey 分别是：AKIDz8krbsJ5yKBZQpn74WFkmLPx3******* 和 Gu5t9xGARNpq86cd98joQYCN3******* 。用户想查看广州区云服务器名为“未命名”的主机状态，只返回一条数据。则请求可能为：

```
curl -X POST https://cvm.tencentcloudapi.com \
-H "Authorization: TC3-HMAC-SHA256 Credential=AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******/2019-02-25/cvm/tc3_request, SignedHeaders=content-type;host, Signature=2230eefd229f582d8b1b891af7107b91597240707d778ab3738f756258d7652c" \
-H "Content-Type: application/json; charset=utf-8" \
-H "Host: cvm.tencentcloudapi.com" \
-H "X-TC-Action: DescribeInstances" \
-H "X-TC-Timestamp: 1551113065" \
-H "X-TC-Version: 2017-03-12" \
-H "X-TC-Region: ap-guangzhou" \
-d '{"Limit": 1, "Filters": [{"Values": ["\u672a\u547d\u540d"], "Name": "instance-name"}]}'
```

下面详细解释签名计算过程。

1. 拼接规范请求串

按如下伪代码格式拼接规范请求串（CanonicalRequest）：

```
CanonicalRequest =
HTTPRequestMethod + '\n' +
CanonicalURI + '\n' +
CanonicalQueryString + '\n' +
CanonicalHeaders + '\n' +
SignedHeaders + '\n' +
HashedRequestPayload
```

字段名称	解释
HTTPRequestMethod	HTTP 请求方法（GET、POST ）。此示例取值为 POST。
CanonicalURI	URI 参数，API 3.0 固定为正斜杠（/）。
CanonicalQueryString	发起 HTTP 请求 URL 中的查询字符串，对于 POST 请求，固定为空字符串""，对于 GET 请求，则为 URL 中问号（?）后面的字符串内容，例如：Limit=10&Offset=0。 注意：CanonicalQueryString 需要参考 RFC3986 进行 URLEncode，字符集 UTF8，推荐使用编程语言标准库，所有特殊字符均需编码，大写形式。

字段名称	解释
CanonicalHeaders	参与签名的头部信息，至少包含 host 和 content-type 两个头部，也可加入自定义的头部参与签名以提高自身请求的唯一性和安全性。 拼接规则： 1. 头部 key 和 value 统一转成小写，并去掉首尾空格，按照 key:value\n 格式拼接； 2. 多个头部，按照头部 key（小写）的 ASCII 升序进行拼接。 此示例计算结果是 content-type:application/json; charset=utf-8\nhost:cvm.tencentcloudapi.com\n。 注意：content-type 必须和实际发送的相符合，有些编程语言网络库即使未指定也会自动添加 charset 值，如果签名时和发送时不一致，服务器会返回签名校验失败。
SignedHeaders	参与签名的头部信息，说明此次请求有哪些头部参与了签名，和 CanonicalHeaders 包含的头部内容是一一对应的。content-type 和 host 为必选头部。 拼接规则： 1. 头部 key 统一转成小写； 2. 多个头部 key（小写）按照 ASCII 升序进行拼接，并且以分号（;）分隔。 此示例为 content-type;host
HashedRequestPayload	请求正文（payload，即 body，此示例为 {"Limit": 1, "Filters": [{"Values": ["\u672a\u547d\u540d"], "Name": "instance-name"}]}）的哈希值，计算伪代码为 Lowercase(HexEncode(Hash.SHA256(RequestPayload)))，即对 HTTP 请求正文做 SHA256 哈希，然后十六进制编码，最后编码串转换成小写字母。对于 GET 请求，RequestPayload 固定为空字符串。此示例计算结果是 35e9c5b0e3ae67532d3c9f17ead6c90222632e5b1ff7f6e89887f1398934f064。

根据以上规则，示例中得到的规范请求串如下：

```
POST
/

content-type:application/json; charset=utf-8
host:cvm.tencentcloudapi.com

content-type;host
35e9c5b0e3ae67532d3c9f17ead6c90222632e5b1ff7f6e89887f1398934f064
```

2. 拼接待签名字符串

按如下格式拼接待签名字符串：

```
StringToSign =
Algorithm + \n +
RequestTimestamp + \n +
CredentialScope + \n +
HashedCanonicalRequest
```

字段名称	解释
Algorithm	签名算法，目前固定为 TC3-HMAC-SHA256。

字段名称	解释
RequestTimestamp	请求时间戳，即请求头部的公共参数 X-TC-Timestamp 取值，取当前时间 UNIX 时间戳，精确到秒。此示例取值为 1551113065。
CredentialScope	凭证范围，格式为 Date/service/tc3_request，包含日期、所请求的服务和终止字符串（tc3_request）。**Date 为 UTC 标准时间的日期，取值需要和公共参数 X-TC-Timestamp 换算的 UTC 标准时间日期一致**；service 为产品名，必须与调用的产品域名一致。此示例计算结果是 2019-02-25/cvm/tc3_request。
HashedCanonicalRequest	前述步骤拼接所得规范请求串的哈希值，计算伪代码为 Lowercase(HexEncode(Hash.SHA256(CanonicalRequest)))。此示例计算结果是 5ffe6a04c0664d6b969fab9a13bdab201d63ee709638e2749d62a09ca18d7031。

注意：

1. **Date 必须从时间戳 X-TC-Timestamp 计算得到，且时区为 UTC+0。如果加入系统本地时区信息，例如东八区，将导致白天和晚上调用成功，但是凌晨时调用必定失败。假设时间戳为 1551113065，在东八区的时间是 2019-02-26 00:44:25，但是计算得到的 Date 取 UTC+0 的日期应为 2019-02-25，而不是 2019-02-26。**
2. **Timestamp 必须是当前系统时间，且需确保系统时间和标准时间是同步的，如果相差超过五分钟则必定失败。如果长时间不和标准时间同步，可能导致运行一段时间后，请求必定失败，返回签名过期错误。**

根据以上规则，示例中得到的待签名字符串如下：

```
TC3-HMAC-SHA256
1551113065
2019-02-25/cvm/tc3_request
5ffe6a04c0664d6b969fab9a13bdab201d63ee709638e2749d62a09ca18d7031
```

3. 计算签名

1）计算派生签名密钥，伪代码如下：

```
SecretKey = "Gu5t9xGARNpq86cd98joQYCN3*******"
SecretDate = HMAC_SHA256("TC3" + SecretKey, Date)
SecretService = HMAC_SHA256(SecretDate, Service)
SecretSigning = HMAC_SHA256(SecretService, "tc3_request")
```

派生出的密钥 SecretDate 、SecretService 和 SecretSigning 是二进制的数据，可能包含不可打印字符，此处不展示中间结果。

请注意，不同的编程语言，HMAC 库函数中参数顺序可能不一样，请以实际情况为准。此处的伪代码密钥参数 key 在前，消息参数 data 在后。通常标准库函数会提供二进制格式的返回值，也可能会提供打印友好的十六进制格式的返回值，此处使用的是二进制格式。

字段名称	解释
SecretKey	原始的 SecretKey，即 Gu5t9xGARNpq86cd98joQYCN3*******。
Date	即 Credential 中的 Date 字段信息。此示例取值为 2019-02-25。
Service	即 Credential 中的 Service 字段信息。此示例取值为 cvm。

2）计算签名，伪代码如下：

```
Signature = HexEncode(HMAC_SHA256(SecretSigning, StringToSign))
```

此示例计算结果是 2230eefd229f582d8b1b891af7107b91597240707d778ab3738f756258d7652c 。

4. 拼接 Authorization

按如下格式拼接 Authorization：

```
Authorization =
Algorithm + ' ' +
'Credential=' + SecretId + '/' + CredentialScope + ', ' +
'SignedHeaders=' + SignedHeaders + ', ' +
'Signature=' + Signature
```

字段名称	解释
Algorithm	签名方法，固定为 TC3-HMAC-SHA256。
SecretId	密钥对中的 SecretId，即 AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******。
CredentialScope	见上文，凭证范围。此示例计算结果是 2019-02-25/cvm/tc3_request。
SignedHeaders	见上文，参与签名的头部信息。此示例取值为 content-type;host。
Signature	签名值。此示例计算结果是 2230eefd229f582d8b1b891af7107b91597240707d778ab3738f756258d7652c。

根据以上规则，示例中得到的值为：

```
TC3-HMAC-SHA256 Credential=AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******/2019-02-25/cvm/tc3_request, SignedHeaders=conte
nt-type;host, Signature=2230eefd229f582d8b1b891af7107b91597240707d778ab3738f756258d7652c
```

最终完整的调用信息如下：

```
POST https://cvm.tencentcloudapi.com/
Authorization: TC3-HMAC-SHA256 Credential=AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******/2019-02-25/cvm/tc3_request, SignedH
eaders=content-type;host, Signature=2230eefd229f582d8b1b891af7107b91597240707d778ab3738f756258d7652c
Content-Type: application/json; charset=utf-8
Host: cvm.tencentcloudapi.com
X-TC-Action: DescribeInstances
X-TC-Version: 2017-03-12
X-TC-Timestamp: 1551113065
X-TC-Region: ap-guangzhou

{"Limit": 1, "Filters": [{"Values": ["\u672a\u547d\u540d"], "Name": "instance-name"}]}
```

签名演示

在实际调用 API 3.0 时，推荐使用配套的腾讯云 SDK 3.0 ，SDK 封装了签名的过程，开发时只关注产品提供的具体接口即可。详细信息参见 SDK 中心。当前支持的编程语言有：

- Python
- Java
- PHP
- Go
- NodeJS
- .NET
- C++

下面提供了不同产品的生成签名 demo，您可以找到对应的产品参考签名的生成：

- Signature Demo

为了更清楚的解释签名过程，下面以实际编程语言为例，将上述的签名过程具体实现。请求的域名、调用的接口和参数的取值都以上述签名过程为准，代码只为解释签名过程，并不具备通用性，实际开发请尽量使用 SDK。

Java

```
import java.nio.charset.Charset;
import java.nio.charset.StandardCharsets;
import java.security.MessageDigest;
import java.text.SimpleDateFormat;
import java.util.Date;
import java.util.TimeZone;
import java.util.TreeMap;
import javax.crypto.Mac;
import javax.crypto.spec.SecretKeySpec;
import javax.xml.bind.DatatypeConverter;

public class TencentCloudAPITC3Demo {
private final static Charset UTF8 = StandardCharsets.UTF_8;
private final static String SECRET_ID = "AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******";
private final static String SECRET_KEY = "Gu5t9xGARNpq86cd98joQYCN3*******";
private final static String CT_JSON = "application/json; charset=utf-8";

public static byte[] hmac256(byte[] key, String msg) throws Exception {
Mac mac = Mac.getInstance("HmacSHA256");
SecretKeySpec secretKeySpec = new SecretKeySpec(key, mac.getAlgorithm());
mac.init(secretKeySpec);
return mac.doFinal(msg.getBytes(UTF8));
}

public static String sha256Hex(String s) throws Exception {
MessageDigest md = MessageDigest.getInstance("SHA-256");
byte[] d = md.digest(s.getBytes(UTF8));
return DatatypeConverter.printHexBinary(d).toLowerCase();
}

public static void main(String[] args) throws Exception {
String service = "cvm";
String host = "cvm.tencentcloudapi.com";
String region = "ap-guangzhou";
```

```
String action = "DescribeInstances";
String version = "2017-03-12";
String algorithm = "TC3-HMAC-SHA256";
String timestamp = "1551113065";
//String timestamp = String.valueOf(System.currentTimeMillis() / 1000);
SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");
// 注意时区，否则容易出错
sdf.setTimeZone(TimeZone.getTimeZone("UTC"));
String date = sdf.format(new Date(Long.valueOf(timestamp + "000")));

// ************ 步骤 1：拼接规范请求串 ************
String httpRequestMethod = "POST";
String canonicalUri = "/";
String canonicalQueryString = "";
String canonicalHeaders = "content-type:application/json; charset=utf-8\n" + "host:" + host + "\n";
String signedHeaders = "content-type;host";

String payload = "{\"Limit\": 1, \"Filters\": [{\"Values\": [\"\\u672a\\u547d\\u540d\"], \"Name\": \"instance-name\"}]}";
String hashedRequestPayload = sha256Hex(payload);
String canonicalRequest = httpRequestMethod + "\n" + canonicalUri + "\n" + canonicalQueryString + "\n"
+ canonicalHeaders + "\n" + signedHeaders + "\n" + hashedRequestPayload;
System.out.println(canonicalRequest);

// ************ 步骤 2：拼接待签名字符串 ************
String credentialScope = date + "/" + service + "/" + "tc3_request";
String hashedCanonicalRequest = sha256Hex(canonicalRequest);
String stringToSign = algorithm + "\n" + timestamp + "\n" + credentialScope + "\n" + hashedCanonicalRequest;
System.out.println(stringToSign);

// ************ 步骤 3：计算签名 ************
byte[] secretDate = hmac256(("TC3" + SECRET_KEY).getBytes(UTF8), date);
byte[] secretService = hmac256(secretDate, service);
byte[] secretSigning = hmac256(secretService, "tc3_request");
String signature = DatatypeConverter.printHexBinary(hmac256(secretSigning, stringToSign)).toLowerCase();
System.out.println(signature);

// ************ 步骤 4：拼接 Authorization ************
String authorization = algorithm + " " + "Credential=" + SECRET_ID + "/" + credentialScope + ", "
+ "SignedHeaders=" + signedHeaders + ", " + "Signature=" + signature;
System.out.println(authorization);

TreeMap<String, String> headers = new TreeMap<String, String>();
headers.put("Authorization", authorization);
headers.put("Content-Type", CT_JSON);
headers.put("Host", host);
headers.put("X-TC-Action", action);
headers.put("X-TC-Timestamp", timestamp);
headers.put("X-TC-Version", version);
headers.put("X-TC-Region", region);
```

```
StringBuilder sb = new StringBuilder();
sb.append("curl -X POST https://").append(host)
.append(" -H \"Authorization: ").append(authorization).append("\"")
.append(" -H \"Content-Type: application/json; charset=utf-8\"")
.append(" -H \"Host: ").append(host).append("\"")
.append(" -H \"X-TC-Action: ").append(action).append("\"")
.append(" -H \"X-TC-Timestamp: ").append(timestamp).append("\"")
.append(" -H \"X-TC-Version: ").append(version).append("\"")
.append(" -H \"X-TC-Region: ").append(region).append("\"")
.append(" -d '").append(payload).append("'");
System.out.println(sb.toString());
}
}
```

Python

```
# -*- coding: utf-8 -*-
import hashlib, hmac, json, os, sys, time
from datetime import datetime

# 密钥参数
secret_id = "AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******"
secret_key = "Gu5t9xGARNpq86cd98joQYCN3*******"

service = "cvm"
host = "cvm.tencentcloudapi.com"
endpoint = "https://" + host
region = "ap-guangzhou"
action = "DescribeInstances"
version = "2017-03-12"
algorithm = "TC3-HMAC-SHA256"
#timestamp = int(time.time())
timestamp = 1551113065
date = datetime.utcfromtimestamp(timestamp).strftime("%Y-%m-%d")
params = {"Limit": 1, "Filters": [{"Name": "instance-name", "Values": [u"未命名"]}]}

# ************* 步骤 1：拼接规范请求串 *************
http_request_method = "POST"
canonical_uri = "/"
canonical_querystring = ""
ct = "application/json; charset=utf-8"
payload = json.dumps(params)
canonical_headers = "content-type:%s\nhost:%s\n" % (ct, host)
signed_headers = "content-type;host"
hashed_request_payload = hashlib.sha256(payload.encode("utf-8")).hexdigest()
canonical_request = (http_request_method + "\n" +
canonical_uri + "\n" +
canonical_querystring + "\n" +
canonical_headers + "\n" +
```

```
signed_headers + "\n" +
hashed_request_payload)
print(canonical_request)

# ************* 步骤 2：拼接待签名字符串 *************
credential_scope = date + "/" + service + "/" + "tc3_request"
hashed_canonical_request = hashlib.sha256(canonical_request.encode("utf-8")).hexdigest()
string_to_sign = (algorithm + "\n" +
str(timestamp) + "\n" +
credential_scope + "\n" +
hashed_canonical_request)
print(string_to_sign)


# ************* 步骤 3：计算签名 *************
# 计算签名摘要函数
def sign(key, msg):
return hmac.new(key, msg.encode("utf-8"), hashlib.sha256).digest()
secret_date = sign(("TC3" + secret_key).encode("utf-8"), date)
secret_service = sign(secret_date, service)
secret_signing = sign(secret_service, "tc3_request")
signature = hmac.new(secret_signing, string_to_sign.encode("utf-8"), hashlib.sha256).hexdigest()
print(signature)

# ************* 步骤 4：拼接 Authorization *************
authorization = (algorithm + " " +
"Credential=" + secret_id + "/" + credential_scope + ", " +
"SignedHeaders=" + signed_headers + ", " +
"Signature=" + signature)
print(authorization)

print('curl -X POST ' + endpoint
+ ' -H "Authorization: ' + authorization + '"'
+ ' -H "Content-Type: application/json; charset=utf-8"'
+ ' -H "Host: ' + host + '"'
+ ' -H "X-TC-Action: ' + action + '"'
+ ' -H "X-TC-Timestamp: ' + str(timestamp) + '"'
+ ' -H "X-TC-Version: ' + version + '"'
+ ' -H "X-TC-Region: ' + region + '"'
+ " -d '" + payload + "'")
```

Golang

```
package main

import (
"crypto/hmac"
"crypto/sha256"
"encoding/hex"
```

```
"fmt"
"time"
)

func sha256hex(s string) string {
b := sha256.Sum256([]byte(s))
return hex.EncodeToString(b[:])
}

func hmacsha256(s, key string) string {
hashed := hmac.New(sha256.New, []byte(key))
hashed.Write([]byte(s))
return string(hashed.Sum(nil))
}

func main() {
secretId := "AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******"
secretKey := "Gu5t9xGARNpq86cd98joQYCN3*******"
host := "cvm.tencentcloudapi.com"
algorithm := "TC3-HMAC-SHA256"
service := "cvm"
version := "2017-03-12"
action := "DescribeInstances"
region := "ap-guangzhou"
//var timestamp int64 = time.Now().Unix()
var timestamp int64 = 1551113065

// step 1: build canonical request string
httpRequestMethod := "POST"
canonicalURI := "/"
canonicalQueryString := ""
canonicalHeaders := "content-type:application/json; charset=utf-8\n" + "host:" + host + "\n"
signedHeaders := "content-type;host"
payload := `{"Limit": 1, "Filters": [{"Values": ["\u672a\u547d\u540d"], "Name": "instance-name"}]}`
hashedRequestPayload := sha256hex(payload)
canonicalRequest := fmt.Sprintf("%s\n%s\n%s\n%s\n%s\n%s",
httpRequestMethod,
canonicalURI,
canonicalQueryString,
canonicalHeaders,
signedHeaders,
hashedRequestPayload)
fmt.Println(canonicalRequest)

// step 2: build string to sign
date := time.Unix(timestamp, 0).UTC().Format("2006-01-02")
credentialScope := fmt.Sprintf("%s/%s/tc3_request", date, service)
hashedCanonicalRequest := sha256hex(canonicalRequest)
string2sign := fmt.Sprintf("%s\n%d\n%s\n%s",
algorithm,
```

```
timestamp,
credentialScope,
hashedCanonicalRequest)
fmt.Println(string2sign)

// step 3: sign string
secretDate := hmacsha256(date, "TC3"+secretKey)
secretService := hmacsha256(service, secretDate)
secretSigning := hmacsha256("tc3_request", secretService)
signature := hex.EncodeToString([]byte(hmacsha256(string2sign, secretSigning)))
fmt.Println(signature)

// step 4: build authorization
authorization := fmt.Sprintf("%s Credential=%s/%s, SignedHeaders=%s, Signature=%s",
algorithm,
secretId,
credentialScope,
signedHeaders,
signature)
fmt.Println(authorization)

curl := fmt.Sprintf(`curl -X POST https://%s\
-H "Authorization: %s"\
-H "Content-Type: application/json; charset=utf-8"\
-H "Host: %s" -H "X-TC-Action: %s"\
-H "X-TC-Timestamp: %d"\
-H "X-TC-Version: %s"\
-H "X-TC-Region: %s"\
-d '%s'`, host, authorization, host, action, timestamp, version, region, payload)
fmt.Println(curl)
}
```

PHP

```
<?php
$secretId = "AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******";
$secretKey = "Gu5t9xGARNpq86cd98joQYCN3*******";
$host = "cvm.tencentcloudapi.com";
$service = "cvm";
$version = "2017-03-12";
$action = "DescribeInstances";
$region = "ap-guangzhou";
// $timestamp = time();
$timestamp = 1551113065;
$algorithm = "TC3-HMAC-SHA256";

// step 1: build canonical request string
$httpRequestMethod = "POST";
$canonicalUri = "/";
```

```
$canonicalQueryString = "";
$canonicalHeaders = "content-type:application/json; charset=utf-8\n"."host:".$host."\n";
$signedHeaders = "content-type;host";
$payload = '{"Limit": 1, "Filters": [{"Values": ["\u672a\u547d\u540d"], "Name": "instance-name"}]}';
$hashedRequestPayload = hash("SHA256", $payload);
$canonicalRequest = $httpRequestMethod."\n"
.$canonicalUri."\n"
.$canonicalQueryString."\n"
.$canonicalHeaders."\n"
.$signedHeaders."\n"
.$hashedRequestPayload;
echo $canonicalRequest.PHP_EOL;

// step 2: build string to sign
$date = gmdate("Y-m-d", $timestamp);
$credentialScope = $date."/".$service."/tc3_request";
$hashedCanonicalRequest = hash("SHA256", $canonicalRequest);
$stringToSign = $algorithm."\n"
.$timestamp."\n"
.$credentialScope."\n"
.$hashedCanonicalRequest;
echo $stringToSign.PHP_EOL;

// step 3: sign string
$secretDate = hash_hmac("SHA256", $date, "TC3".$secretKey, true);
$secretService = hash_hmac("SHA256", $service, $secretDate, true);
$secretSigning = hash_hmac("SHA256", "tc3_request", $secretService, true);
$signature = hash_hmac("SHA256", $stringToSign, $secretSigning);
echo $signature.PHP_EOL;

// step 4: build authorization
$authorization = $algorithm
." Credential=".$secretId."/".$credentialScope
.", SignedHeaders=content-type;host, Signature=".$signature;
echo $authorization.PHP_EOL;

$curl = "curl -X POST https://".$host
.' -H "Authorization: '.$authorization.'"'
.' -H "Content-Type: application/json; charset=utf-8"'
.' -H "Host: '.$host.'"'
.' -H "X-TC-Action: '.$action.'"'
.' -H "X-TC-Timestamp: '.$timestamp.'"'
.' -H "X-TC-Version: '.$version.'"'
.' -H "X-TC-Region: '.$region.'"'
." -d '".$payload."'";
echo $curl.PHP_EOL;
```

Ruby

```
# -*- coding: UTF-8 -*-
# require ruby>=2.3.0
require 'digest'
require 'json'
require 'time'
require 'openssl'

# 密钥参数
secret_id = 'AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******'
secret_key = 'Gu5t9xGARNpq86cd98joQYCN3*******'

service = 'cvm'
host = 'cvm.tencentcloudapi.com'
endpoint = 'https://' + host
region = 'ap-guangzhou'
action = 'DescribeInstances'
version = '2017-03-12'
algorithm = 'TC3-HMAC-SHA256'
# timestamp = Time.now.to_i
timestamp = 1551113065
date = Time.at(timestamp).utc.strftime('%Y-%m-%d')

# ************* 步骤 1：拼接规范请求串 *************
http_request_method = 'POST'
canonical_uri = '/'
canonical_querystring = ''
canonical_headers = "content-type:application/json; charset=utf-8\nhost:#{host}\n"
signed_headers = 'content-type;host'
# params = { 'Limit' => 1, 'Filters' => [{ 'Name' => 'instance-name', 'Values' => ['未命名'] }] }
# payload = JSON.generate(params, { 'ascii_only' => true, 'space' => ' ' })
# json will generate in random order, to get specified result in example, we hard-code it here.
payload = '{"Limit": 1, "Filters": [{"Values": ["\u672a\u547d\u540d"], "Name": "instance-name"}]}'
hashed_request_payload = Digest::SHA256.hexdigest(payload)
canonical_request = [
http_request_method,
canonical_uri,
canonical_querystring,
canonical_headers,
signed_headers,
hashed_request_payload,
].join("\n")

puts canonical_request

# ************* 步骤 2：拼接待签名字符串 *************
credential_scope = date + '/' + service + '/' + 'tc3_request'
hashed_request_payload = Digest::SHA256.hexdigest(canonical_request)
string_to_sign = [
algorithm,
```

```
timestamp.to_s,
credential_scope,
hashed_request_payload,
].join("\n")
puts string_to_sign

# ************* 步骤 3：计算签名 *************
digest = OpenSSL::Digest.new('sha256')
secret_date = OpenSSL::HMAC.digest(digest, 'TC3' + secret_key, date)
secret_service = OpenSSL::HMAC.digest(digest, secret_date, service)
secret_signing = OpenSSL::HMAC.digest(digest, secret_service, 'tc3_request')
signature = OpenSSL::HMAC.hexdigest(digest, secret_signing, string_to_sign)
puts signature

# ************* 步骤 4：拼接 Authorization *************
authorization = "#{algorithm} Credential=#{secret_id}/#{credential_scope}, SignedHeaders=#{signed_headers}, Signature=
#{signature}"
puts authorization

puts 'curl -X POST ' + endpoint + \
+ ' -H "Authorization: ' + authorization + '"' \
+ ' -H "Content-Type: application/json; charset=utf-8"' \
+ ' -H "Host: ' + host + '"' \
+ ' -H "X-TC-Action: ' + action + '"' \
+ ' -H "X-TC-Timestamp: ' + timestamp.to_s + '"' \
+ ' -H "X-TC-Version: ' + version + '"' \
+ ' -H "X-TC-Region: ' + region + '"' \
+ " -d '" + payload + "'"
```

DotNet

```
using System;
using System.Collections.Generic;
using System.Security.Cryptography;
using System.Text;

public class Application
{
public static string SHA256Hex(string s)
{
using (SHA256 algo = SHA256.Create())
{
byte[] hashbytes = algo.ComputeHash(Encoding.UTF8.GetBytes(s));
StringBuilder builder = new StringBuilder();
for (int i = 0; i < hashbytes.Length; ++i)
{
builder.Append(hashbytes[i].ToString("x2"));
}
return builder.ToString();
```

```
}
}

public static byte[] HmacSHA256(byte[] key, byte[] msg)
{
using (HMACSHA256 mac = new HMACSHA256(key))
{
return mac.ComputeHash(msg);
}
}

public static Dictionary<String, String> BuildHeaders(string secretid,
string secretkey, string service, string endpoint, string region,
string action, string version, DateTime date, string requestPayload)
{
string datestr = date.ToString("yyyy-MM-dd");
DateTime startTime = new DateTime(1970, 1, 1, 0, 0, 0, 0, DateTimeKind.Utc);
long requestTimestamp = (long)Math.Round((date - startTime).TotalMilliseconds, MidpointRounding.AwayFromZero) / 1000;
// ************* 步骤 1：拼接规范请求串 *************
string algorithm = "TC3-HMAC-SHA256";
string httpRequestMethod = "POST";
string canonicalUri = "/";
string canonicalQueryString = "";
string contentType = "application/json";
string canonicalHeaders = "content-type:" + contentType + "; charset=utf-8\n" + "host:" + endpoint + "\n";
string signedHeaders = "content-type;host";
string hashedRequestPayload = SHA256Hex(requestPayload);
string canonicalRequest = httpRequestMethod + "\n"
+ canonicalUri + "\n"
+ canonicalQueryString + "\n"
+ canonicalHeaders + "\n"
+ signedHeaders + "\n"
+ hashedRequestPayload;
Console.WriteLine(canonicalRequest);

// ************* 步骤 2：拼接待签名字符串 *************
string credentialScope = datestr + "/" + service + "/" + "tc3_request";
string hashedCanonicalRequest = SHA256Hex(canonicalRequest);
string stringToSign = algorithm + "\n" + requestTimestamp.ToString() + "\n" + credentialScope + "\n" + hashedCanonicalReque
st;
Console.WriteLine(stringToSign);

// ************* 步骤 3：计算签名 *************
byte[] tc3SecretKey = Encoding.UTF8.GetBytes("TC3" + secretkey);
byte[] secretDate = HmacSHA256(tc3SecretKey, Encoding.UTF8.GetBytes(datestr));
byte[] secretService = HmacSHA256(secretDate, Encoding.UTF8.GetBytes(service));
byte[] secretSigning = HmacSHA256(secretService, Encoding.UTF8.GetBytes("tc3_request"));
byte[] signatureBytes = HmacSHA256(secretSigning, Encoding.UTF8.GetBytes(stringToSign));
string signature = BitConverter.ToString(signatureBytes).Replace("-", "").ToLower();
Console.WriteLine(signature);
```

```
// ************* 步骤 4：拼接 Authorization *************
string authorization = algorithm + " "
+ "Credential=" + secretid + "/" + credentialScope + ", "
+ "SignedHeaders=" + signedHeaders + ", "
+ "Signature=" + signature;
Console.WriteLine(authorization);

Dictionary<string, string> headers = new Dictionary<string, string>();
headers.Add("Authorization", authorization);
headers.Add("Host", endpoint);
headers.Add("Content-Type", contentType + "; charset=utf-8");
headers.Add("X-TC-Timestamp", requestTimestamp.ToString());
headers.Add("X-TC-Version", version);
headers.Add("X-TC-Action", action);
headers.Add("X-TC-Region", region);
return headers;
}
public static void Main(string[] args)
{
// 密钥参数
string SECRET_ID = "AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******";
string SECRET_KEY = "Gu5t9xGARNpq86cd98joQYCN3*******";

string service = "cvm";
string endpoint = "cvm.tencentcloudapi.com";
string region = "ap-guangzhou";
string action = "DescribeInstances";
string version = "2017-03-12";

// 此处由于示例规范的原因，采用时间戳2019-02-26 00:44:25，此参数作为示例，如果在项目中，您应当使用：
// DateTime date = DateTime.UtcNow;
// 注意时区，建议此时间统一采用UTC时间戳，否则容易出错
DateTime date = new DateTime(1970, 1, 1, 0, 0, 0, 0, DateTimeKind.Utc).AddSeconds(1551113065);
string requestPayload = "{\"Limit\": 1, \"Filters\": [{\"Values\": [\"\\u672a\\u547d\\u540d\"], \"Name\": \"instance-name\"}]}";

Dictionary<string, string> headers = BuildHeaders(SECRET_ID, SECRET_KEY, service
, endpoint, region, action, version, date, requestPayload);

Console.WriteLine("POST https://cvm.tencentcloudapi.com");
foreach (KeyValuePair<string, string> kv in headers)
{
Console.WriteLine(kv.Key + ": " + kv.Value);
}
Console.WriteLine();
Console.WriteLine(requestPayload);
}
}
```

NodeJS

```
const crypto = require('crypto');

function sha256(message, secret = '', encoding) {
const hmac = crypto.createHmac('sha256', secret)
return hmac.update(message).digest(encoding)
}

function getHash(message, encoding = 'hex') {
const hash = crypto.createHash('sha256')
return hash.update(message).digest(encoding)
}

function getDate(timestamp) {
const date = new Date(timestamp * 1000)
const year = date.getUTCFullYear()
const month = ('0' + (date.getUTCMonth() + 1)).slice(-2)
const day = ('0' + date.getUTCDate()).slice(-2)
return `${year}-${month}-${day}`
}

function main(){
// 密钥参数
const SECRET_ID = "AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******"
const SECRET_KEY = "Gu5t9xGARNpq86cd98joQYCN3*******"

const endpoint = "cvm.tencentcloudapi.com"
const service = "cvm"
const region = "ap-guangzhou"
const action = "DescribeInstances"
const version = "2017-03-12"
//const timestamp = getTime()
const timestamp = 1551113065
//时间处理, 获取世界时间日期
const date = getDate(timestamp)

// ************* 步骤 1：拼接规范请求串 *************
const signedHeaders = "content-type;host"

const payload = "{\"Limit\": 1, \"Filters\": [{\"Values\": [\"\\u672a\\u547d\\u540d\"], \"Name\": \"instance-name\"}]}"

const hashedRequestPayload = getHash(payload);
const httpRequestMethod = "POST"
const canonicalUri = "/"
const canonicalQueryString = ""
const canonicalHeaders = "content-type:application/json; charset=utf-8\n" + "host:" + endpoint + "\n"

const canonicalRequest = httpRequestMethod + "\n"
+ canonicalUri + "\n"
```

```
+ canonicalQueryString + "\n"
+ canonicalHeaders + "\n"
+ signedHeaders + "\n"
+ hashedRequestPayload
console.log(canonicalRequest)

// ************* 步骤 2：拼接待签名字符串 *************
const algorithm = "TC3-HMAC-SHA256"
const hashedCanonicalRequest = getHash(canonicalRequest);
const credentialScope = date + "/" + service + "/" + "tc3_request"
const stringToSign = algorithm + "\n" +
timestamp + "\n" +
credentialScope + "\n" +
hashedCanonicalRequest
console.log(stringToSign)

// ************* 步骤 3：计算签名 *************
const kDate = sha256(date, 'TC3' + SECRET_KEY)
const kService = sha256(service, kDate)
const kSigning = sha256('tc3_request', kService)
const signature = sha256(stringToSign, kSigning, 'hex')
console.log(signature)

// ************* 步骤 4：拼接 Authorization *************
const authorization = algorithm + " " +
"Credential=" + SECRET_ID + "/" + credentialScope + ", " +
"SignedHeaders=" + signedHeaders + ", " +
"Signature=" + signature
console.log(authorization)

const curlcmd = 'curl -X POST ' + "https://" + endpoint
+ ' -H "Authorization: ' + authorization + '"'
+ ' -H "Content-Type: application/json; charset=utf-8"'
+ ' -H "Host: ' + endpoint + '"'
+ ' -H "X-TC-Action: ' + action + '"'
+ ' -H "X-TC-Timestamp: ' + timestamp.toString() + '"'
+ ' -H "X-TC-Version: ' + version + '"'
+ ' -H "X-TC-Region: ' + region + '"'
+ " -d '" + payload + "'"
console.log(curlcmd)
}
main()
```

C++

```
#include <iostream>
#include <iomanip>
#include <sstream>
#include <string>
```

```
#include <stdio.h>
#include <time.h>
#include <openssl/sha.h>
#include <openssl/hmac.h>

using namespace std;

string get_data(int64_t &timestamp)
{
string utcDate;
char buff[20] = {0};
// time_t timenow;
struct tm sttime;
sttime = *gmtime(&timestamp);
strftime(buff, sizeof(buff), "%Y-%m-%d", &sttime);
utcDate = string(buff);
return utcDate;
}

string int2str(int64_t n)
{
std::stringstream ss;
ss << n;
return ss.str();
}

string sha256Hex(const string &str)
{
char buf[3];
unsigned char hash[SHA256_DIGEST_LENGTH];
SHA256_CTX sha256;
SHA256_Init(&sha256);
SHA256_Update(&sha256, str.c_str(), str.size());
SHA256_Final(hash, &sha256);
std::string NewString = "";
for(int i = 0; i < SHA256_DIGEST_LENGTH; i++)
{
snprintf(buf, sizeof(buf), "%02x", hash[i]);
NewString = NewString + buf;
}
return NewString;
}

string HmacSha256(const string &key, const string &input)
{
unsigned char hash[32];

HMAC_CTX *h;
#if OPENSSL_VERSION_NUMBER < 0x10100000L
HMAC_CTX hmac;
```

```
HMAC_CTX_init(&hmac);
h = &hmac;
#else
h = HMAC_CTX_new();
#endif

HMAC_Init_ex(h, &key[0], key.length(), EVP_sha256(), NULL);
HMAC_Update(h, ( unsigned char* )&input[0], input.length());
unsigned int len = 32;
HMAC_Final(h, hash, &len);

#if OPENSSL_VERSION_NUMBER < 0x10100000L
HMAC_CTX_cleanup(h);
#else
HMAC_CTX_free(h);
#endif

std::stringstream ss;
ss << std::setfill('0');
for (int i = 0; i < len; i++)
{
ss << hash[i];
}

return (ss.str());
}

string HexEncode(const string &input)
{
static const char* const lut = "0123456789abcdef";
size_t len = input.length();

string output;
output.reserve(2 * len);
for (size_t i = 0; i < len; ++i)
{
const unsigned char c = input[i];
output.push_back(lut[c >> 4]);
output.push_back(lut[c & 15]);
}
return output;
}

int main()
{
// 密钥参数
string SECRET_ID = "AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******";
string SECRET_KEY = "Gu5t9xGARNpq86cd98joQYCN3*******";

string service = "cvm";
```

```
string host = "cvm.tencentcloudapi.com";
string region = "ap-guangzhou";
string action = "DescribeInstances";
string version = "2017-03-12";
int64_t timestamp = 1551113065;
string date = get_data(timestamp);

// ************* 步骤 1：拼接规范请求串 *************
string httpRequestMethod = "POST";
string canonicalUri = "/";
string canonicalQueryString = "";
string canonicalHeaders = "content-type:application/json; charset=utf-8\nhost:" + host + "\n";
string signedHeaders = "content-type;host";
string payload = "{\"Limit\": 1, \"Filters\": [{\"Values\": [\"\\u672a\\u547d\\u540d\"], \"Name\": \"instance-name\"}]}";
string hashedRequestPayload = sha256Hex(payload);
string canonicalRequest = httpRequestMethod + "\n" + canonicalUri + "\n" + canonicalQueryString + "\n"
+ canonicalHeaders + "\n" + signedHeaders + "\n" + hashedRequestPayload;
cout << canonicalRequest << endl;

// ************* 步骤 2：拼接待签名字符串 *************
string algorithm = "TC3-HMAC-SHA256";
string RequestTimestamp = int2str(timestamp);
string credentialScope = date + "/" + service + "/" + "tc3_request";
string hashedCanonicalRequest = sha256Hex(canonicalRequest);
string stringToSign = algorithm + "\n" + RequestTimestamp + "\n" + credentialScope + "\n" + hashedCanonicalRequest;
cout << stringToSign << endl;

// ************* 步骤 3：计算签名 ***************
string kKey = "TC3" + SECRET_KEY;
string kDate = HmacSha256(kKey, date);
string kService = HmacSha256(kDate, service);
string kSigning = HmacSha256(kService, "tc3_request");
string signature = HexEncode(HmacSha256(kSigning, stringToSign));
cout << signature << endl;

// ************* 步骤 4：拼接 Authorization *************
string authorization = algorithm + " " + "Credential=" + SECRET_ID + "/" + credentialScope + ", "
+ "SignedHeaders=" + signedHeaders + ", " + "Signature=" + signature;
cout << authorization << endl;

string curlcmd = "curl -X POST https://" + host + "\n"
+ " -H \"Authorization: " + authorization + "\n"
+ " -H \"Content-Type: application/json; charset=utf-8\"" + "\n"
+ " -H \"Host: " + host + "\n"
+ " -H \"X-TC-Action: " + action + "\n"
+ " -H \"X-TC-Timestamp: " + RequestTimestamp + "\n"
+ " -H \"X-TC-Version: " + version + "\n"
+ " -H \"X-TC-Region: " + region + "\n"
+ " -d '" + payload;
cout << curlcmd << endl;
```

```
return 0;
};
```

签名失败

存在以下签名失败的错误码，请根据实际情况处理。

错误码	错误描述
AuthFailure.SignatureExpire	签名过期。Timestamp 与服务器接收到请求的时间相差不得超过五分钟。
AuthFailure.SecretIdNotFound	密钥不存在。请到控制台查看密钥是否被禁用，是否少复制了字符或者多了字符。
AuthFailure.SignatureFailure	签名错误。可能是签名计算错误，或者签名与实际发送的内容不相符合，也有可能是密钥 SecretKey 错误导致的。
AuthFailure.TokenFailure	临时证书 Token 错误。
AuthFailure.InvalidSecretId	密钥非法（不是云 API 密钥类型）。

签名方法

最近更新时间：2022-04-12 06:15:03

签名方法 v1 简单易用，但是功能和安全性都不如签名方法 v3，推荐使用签名方法 v3。

首次接触，建议使用 API Explorer 中的“签名串生成”功能，选择签名版本为“API 3.0 签名 v1”，可以生成签名过程进行验证，并提供了部分编程语言的签名示例，也可直接生成 SDK 代码。推荐使用腾讯云 API 配套的 7 种常见的编程语言 SDK，已经封装了签名和请求过程，均已开源，支持 Python、Java、PHP、Go、NodeJS、.NET、C++。

推荐使用 API Explorer 点击调试

您可以通过 API Explorer 的【签名串生成】模块查看每个接口签名的生成过程。

腾讯云 API 会对每个访问请求进行身份验证，即每个请求都需要在公共请求参数中包含签名信息（Signature）以验证请求者身份。
签名信息由安全凭证生成，安全凭证包括 SecretId 和 SecretKey；若用户还没有安全凭证，请前往 云API密钥页面 申请，否则无法调用云 API 接口。

1. 申请安全凭证

在第一次使用云 API 之前，请前往 云 API 密钥页面 申请安全凭证。
安全凭证包括 SecretId 和 SecretKey：

- SecretId 用于标识 API 调用者身份
- SecretKey 用于加密签名字符串和服务器端验证签名字符串的密钥。
- **用户必须严格保管安全凭证，避免泄露。**

申请安全凭证的具体步骤如下：

1. 登录 腾讯云管理中心控制台 。
2. 前往 云 API 密钥 的控制台页面
3. 在 云 API 密钥 页面，单击【新建密钥】即可以创建一对 SecretId/SecretKey。

注意：每个账号最多可以拥有两对 SecretId/SecretKey。

2. 生成签名串

有了安全凭证SecretId 和 SecretKey后，就可以生成签名串了。以下是使用签名方法 v1 生成签名串的详细过程：

假设用户的 SecretId 和 SecretKey 分别是：

- SecretId: AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******
- SecretKey: Gu5t9xGARNpq86cd98joQYCN3*******

注意：这里只是示例，请根据用户实际申请的 SecretId 和 SecretKey 进行后续操作！

以云服务器查看实例列表(DescribeInstances)请求为例，当用户调用这一接口时，其请求参数可能如下：

参数名称	中文	参数值
Action	方法名	DescribeInstances
SecretId	密钥 ID	AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******
Timestamp	当前时间戳	1465185768
Nonce	随机正整数	11886

参数名称	中文	参数值
Region	实例所在区域	ap-guangzhou
InstanceIds.0	待查询的实例 ID	ins-09dx96dg
Offset	偏移量	0
Limit	最大允许输出	20
Version	接口版本号	2017-03-12

这里只展示了部分公共参数和接口输入参数，用户可以根据实际需要添加其他参数，例如 Language 和 Token 公共参数。

2.1. 对参数排序

首先对所有请求参数按参数名的字典序（ ASCII 码）升序排序。注意：1）只按参数名进行排序，参数值保持对应即可，不参与比大小；2）按 ASCII 码比大小，如 InstanceIds.2 要排在 InstanceIds.12 后面，不是按字母表，也不是按数值。用户可以借助编程语言中的相关排序函数来实现这一功能，如 PHP 中的 ksort 函数。上述示例参数的排序结果如下:

```
{
'Action' : 'DescribeInstances',
'InstanceIds.0' : 'ins-09dx96dg',
'Limit' : 20,
'Nonce' : 11886,
'Offset' : 0,
'Region' : 'ap-guangzhou',
'SecretId' : 'AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******',
'Timestamp' : 1465185768,
'Version': '2017-03-12',
}
```

使用其它程序设计语言开发时，可对上面示例中的参数进行排序，得到的结果一致即可。

2.2. 拼接请求字符串

此步骤生成请求字符串。
将把上一步排序好的请求参数格式化成“参数名称=参数值”的形式，如对 Action 参数，其参数名称为 "Action" ，参数值为 "DescribeInstances" ，因此格式化后就为 Action=DescribeInstances 。
注意：“参数值”为原始值而非 url 编码后的值。

然后将格式化后的各个参数用"&"拼接在一起，最终生成的请求字符串为:

```
Action=DescribeInstances&InstanceIds.0=ins-09dx96dg&Limit=20&Nonce=11886&Offset=0&Region=ap-guangzhou&SecretId=AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******&Timestamp=1465185768&Version=2017-03-12
```

2.3. 拼接签名原文字符串

此步骤生成签名原文字符串。
签名原文字符串由以下几个参数构成:

1. 请求方法: 支持 POST 和 GET 方式，这里使用 GET 请求，注意方法为全大写。
2. 请求主机:查看实例列表(DescribeInstances)的请求域名为：cvm.tencentcloudapi.com。实际的请求域名根据接口所属模块的不同而不同，详见各接口说明。
3. 请求路径: 当前版本云API的请求路径固定为 / 。

4. 请求字符串: 即上一步生成的请求字符串。

签名原文串的拼接规则为： 请求方法 + 请求主机 +请求路径 + ? + 请求字符串 。

示例的拼接结果为：

```
GETcvm.tencentcloudapi.com/?Action=DescribeInstances&InstanceIds.0=ins-09dx96dg&Limit=20&Nonce=11886&Offset=0&Region=ap-guangzhou&SecretId=AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******&Timestamp=1465185768&Version=2017-03-12
```

2.4. 生成签名串

此步骤生成签名串。
首先使用 HMAC-SHA1 算法对上一步中获得的**签名原文字符串**进行签名，然后将生成的签名串使用 Base64 进行编码，即可获得最终的签名串。

具体代码如下，以 PHP 语言为例：

```
$secretKey = 'Gu5t9xGARNpq86cd98joQYCN3*******';
$srcStr = 'GETcvm.tencentcloudapi.com/?Action=DescribeInstances&InstanceIds.0=ins-09dx96dg&Limit=20&Nonce=11886&Offset=0&Region=ap-guangzhou&SecretId=AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******&Timestamp=1465185768&Version=2017-03-12';
$signStr = base64_encode(hash_hmac('sha1', $srcStr, $secretKey, true));
echo $signStr;
```

最终得到的签名串为：

```
zmmjn35mikh6pM3V7sUEuX4wyYM=
```

使用其它程序设计语言开发时，可用上面示例中的原文进行签名验证，得到的签名串与例子中的一致即可。

3. 签名串编码

生成的签名串并不能直接作为请求参数，需要对其进行 URL 编码。

如上一步生成的签名串为 zmmjn35mikh6pM3V7sUEuX4wyYM= ，最终得到的签名串请求参数（Signature）为：zmmjn35mikh6pM3V7sUEuX4wyYM%3D，它将用于生成最终的请求 URL。

注意：如果用户的请求方法是 GET，或者请求方法为 POST 同时 Content-Type 为 application/x-www-form-urlencoded，则发送请求时所有请求参数的值均需要做 URL 编码，参数键和=符号不需要编码。非 ASCII 字符在 URL 编码前需要先以 UTF-8 进行编码。

注意：有些编程语言的网络库会自动为所有参数进行 urlencode，在这种情况下，就不需要对签名串进行 URL 编码了，否则两次 URL 编码会导致签名失败。

注意：其他参数值也需要进行编码，编码采用 RFC 3986。使用 %XY 对特殊字符例如汉字进行百分比编码，其中“X”和“Y”为十六进制字符（0-9 和大写字母 A-F），使用小写将引发错误。

4. 签名失败

根据实际情况，存在以下签名失败的错误码，请根据实际情况处理。

错误代码	错误描述
AuthFailure.SignatureExpire	签名过期
AuthFailure.SecretIdNotFound	密钥不存在

错误代码	错误描述
AuthFailure.SignatureFailure	签名错误
AuthFailure.TokenFailure	token 错误
AuthFailure.InvalidSecretId	密钥非法（不是云 API 密钥类型）

5. 签名演示

在实际调用 API 3.0 时，推荐使用配套的腾讯云 SDK 3.0，SDK 封装了签名的过程，开发时只关注产品提供的具体接口即可。详细信息参见 SDK 中心。当前支持的编程语言有：

- Python
- Java
- PHP
- Go
- NodeJS
- .NET
- C++

下面提供了不同产品的生成签名 demo，您可以找到对应的产品参考签名的生成：

- Signature Demo

为了更清楚的解释签名过程，下面以实际编程语言为例，将上述的签名过程具体实现。请求的域名、调用的接口和参数的取值都以上述签名过程为准，代码只为解释签名过程，并不具备通用性，实际开发请尽量使用 SDK 。

最终输出的 url 可能为：https://cvm.tencentcloudapi.com/?Action=DescribeInstances&InstanceIds.0=ins-09dx96dg&Limit=20&Nonce=11886&Offset=0&Region=ap-guangzhou&SecretId=AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******&Signature=zmmjn35mikh6pM3V7sUEuX4wyYM%3D&Timestamp=1465185768&Version=2017-03-12 。

注意：由于示例中的密钥是虚构的，时间戳也不是系统当前时间，因此如果将此 url 在浏览器中打开或者用 curl 等命令调用时会返回鉴权错误：签名过期。为了得到一个可以正常返回的 url，需要修改示例中的 SecretId 和 SecretKey 为真实的密钥，并使用系统当前时间戳作为 Timestamp 。

注意：在下面的示例中，不同编程语言，甚至同一语言每次执行得到的 url 可能都有所不同，表现为参数的顺序不同，但这并不影响正确性。只要所有参数都在，且签名计算正确即可。

注意：以下代码仅适用于 API 3.0，不能直接用于其他的签名流程，即使是旧版的 API，由于存在细节差异也会导致签名计算错误，请以对应的实际文档为准。

Java

```
import java.io.UnsupportedEncodingException;
import java.net.URLEncoder;
import java.util.Random;
import java.util.TreeMap;
import javax.crypto.Mac;
import javax.crypto.spec.SecretKeySpec;
import javax.xml.bind.DatatypeConverter;

public class TencentCloudAPIDemo {
private final static String CHARSET = "UTF-8";
```

```
public static String sign(String s, String key, String method) throws Exception {
Mac mac = Mac.getInstance(method);
SecretKeySpec secretKeySpec = new SecretKeySpec(key.getBytes(CHARSET), mac.getAlgorithm());
mac.init(secretKeySpec);
byte[] hash = mac.doFinal(s.getBytes(CHARSET));
return DatatypeConverter.printBase64Binary(hash);
}

public static String getStringToSign(TreeMap<String, Object> params) {
StringBuilder s2s = new StringBuilder("GETcvm.tencentcloudapi.com/?");
// 签名时要求对参数进行字典排序，此处用TreeMap保证顺序
for (String k : params.keySet()) {
s2s.append(k).append("=").append(params.get(k).toString()).append("&");
}
return s2s.toString().substring(0, s2s.length() - 1);
}

public static String getUrl(TreeMap<String, Object> params) throws UnsupportedEncodingException {
StringBuilder url = new StringBuilder("https://cvm.tencentcloudapi.com/?");
// 实际请求的url中对参数顺序没有要求
for (String k : params.keySet()) {
// 需要对请求串进行urlencode，由于key都是英文字母，故此处仅对其value进行urlencode
url.append(k).append("=").append(URLEncoder.encode(params.get(k).toString(), CHARSET)).append("&");
}
return url.toString().substring(0, url.length() - 1);
}

public static void main(String[] args) throws Exception {
TreeMap<String, Object> params = new TreeMap<String, Object>(); // TreeMap可以自动排序
// 实际调用时应当使用随机数，例如：params.put("Nonce", new Random().nextInt(java.lang.Integer.MAX_VALUE));
params.put("Nonce", 11886); // 公共参数
// 实际调用时应当使用系统当前时间，例如： params.put("Timestamp", System.currentTimeMillis() / 1000);
params.put("Timestamp", 1465185768); // 公共参数
params.put("SecretId", "AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******"); // 公共参数
params.put("Action", "DescribeInstances"); // 公共参数
params.put("Version", "2017-03-12"); // 公共参数
params.put("Region", "ap-guangzhou"); // 公共参数
params.put("Limit", 20); // 业务参数
params.put("Offset", 0); // 业务参数
params.put("InstanceIds.0", "ins-09dx96dg"); // 业务参数
params.put("Signature", sign(getStringToSign(params), "Gu5t9xGARNpq86cd98joQYCN3*******", "HmacSHA1")); // 公共参数
System.out.println(getUrl(params));
}
}
```

Python

注意：如果是在 Python 2 环境中运行，需要先安装 requests 依赖包： pip install requests 。

```python
# -*- coding: utf8 -*-
import base64
import hashlib
import hmac
import time

import requests

secret_id = "AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******"
secret_key = "Gu5t9xGARNpq86cd98joQYCN3*******"

def get_string_to_sign(method, endpoint, params):
s = method + endpoint + "/?"
query_str = "&".join("%s=%s" % (k, params[k]) for k in sorted(params))
return s + query_str

def sign_str(key, s, method):
hmac_str = hmac.new(key.encode("utf8"), s.encode("utf8"), method).digest()
return base64.b64encode(hmac_str)

if __name__ == '__main__':
endpoint = "cvm.tencentcloudapi.com"
data = {
'Action' : 'DescribeInstances',
'InstanceIds.0' : 'ins-09dx96dg',
'Limit' : 20,
'Nonce' : 11886,
'Offset' : 0,
'Region' : 'ap-guangzhou',
'SecretId' : secret_id,
'Timestamp' : 1465185768, # int(time.time())
'Version': '2017-03-12'
}
s = get_string_to_sign("GET", endpoint, data)
data["Signature"] = sign_str(secret_key, s, hashlib.sha1)
print(data["Signature"])
# 此处会实际调用，成功后可能产生计费
# resp = requests.get("https://" + endpoint, params=data)
# print(resp.url)
```

Golang

```golang
package main

import (
"bytes"
"crypto/hmac"
"crypto/sha1"
"encoding/base64"
```

```
"fmt"
"sort"
"strconv"
)

func main() {
secretId := "AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******"
secretKey := "Gu5t9xGARNpq86cd98joQYCN3*******"
params := map[string]string{
"Nonce": "11886",
"Timestamp": strconv.Itoa(1465185768),
"Region": "ap-guangzhou",
"SecretId": secretId,
"Version": "2017-03-12",
"Action": "DescribeInstances",
"InstanceIds.0": "ins-09dx96dg",
"Limit": strconv.Itoa(20),
"Offset": strconv.Itoa(0),
}

var buf bytes.Buffer
buf.WriteString("GET")
buf.WriteString("cvm.tencentcloudapi.com")
buf.WriteString("/")
buf.WriteString("?")

// sort keys by ascii asc order
keys := make([]string, 0, len(params))
for k, _ := range params {
keys = append(keys, k)
}
sort.Strings(keys)

for i := range keys {
k := keys[i]
buf.WriteString(k)
buf.WriteString("=")
buf.WriteString(params[k])
buf.WriteString("&")
}
buf.Truncate(buf.Len() - 1)

hashed := hmac.New(sha1.New, []byte(secretKey))
hashed.Write(buf.Bytes())

fmt.Println(base64.StdEncoding.EncodeToString(hashed.Sum(nil)))
}
```

PHP

```
<?php
$secretId = "AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******";
$secretKey = "Gu5t9xGARNpq86cd98joQYCN3*******";
$param["Nonce"] = 11886;//rand();
$param["Timestamp"] = 1465185768;//time();
$param["Region"] = "ap-guangzhou";
$param["SecretId"] = $secretId;
$param["Version"] = "2017-03-12";
$param["Action"] = "DescribeInstances";
$param["InstanceIds.0"] = "ins-09dx96dg";
$param["Limit"] = 20;
$param["Offset"] = 0;

ksort($param);

$signStr = "GETcvm.tencentcloudapi.com/?";
foreach ( $param as $key => $value ) {
$signStr = $signStr . $key . "=" . $value . "&";
}
$signStr = substr($signStr, 0, -1);

$signature = base64_encode(hash_hmac("sha1", $signStr, $secretKey, true));
echo $signature.PHP_EOL;
// need to install and enable curl extension in php.ini
// $param["Signature"] = $signature;
// $url = "https://cvm.tencentcloudapi.com/?".http_build_query($param);
// echo $url.PHP_EOL;
// $ch = curl_init();
// curl_setopt($ch, CURLOPT_URL, $url);
// $output = curl_exec($ch);
// curl_close($ch);
// echo json_decode($output);
```

Ruby

```
# -*- coding: UTF-8 -*-
# require ruby>=2.3.0
require 'time'
require 'openssl'
require 'base64'

secret_id = "AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******"
secret_key = "Gu5t9xGARNpq86cd98joQYCN3*******"

method = 'GET'
endpoint = 'cvm.tencentcloudapi.com'
data = {
'Action' => 'DescribeInstances',
'InstanceIds.0' => 'ins-09dx96dg',
```

```
'Limit' => 20,
'Nonce' => 11886,
'Offset' => 0,
'Region' => 'ap-guangzhou',
'SecretId' => secret_id,
'Timestamp' => 1465185768, # Time.now.to_i
'Version' => '2017-03-12',
}
sign = method + endpoint + '/?'
params = []
data.sort.each do |item|
params << "#{item[0]}=#{item[1]}"
end
sign += params.join('&')
digest = OpenSSL::Digest.new('sha1')
data['Signature'] = Base64.encode64(OpenSSL::HMAC.digest(digest, secret_key, sign))
puts data['Signature']

# require 'net/http'
# uri = URI('https://' + endpoint)
# uri.query = URI.encode_www_form(data)
# p uri
# res = Net::HTTP.get_response(uri)
# puts res.body
```

DotNet

```
using System;
using System.Collections.Generic;
using System.Net;
using System.Security.Cryptography;
using System.Text;

public class Application {
public static string Sign(string signKey, string secret)
{
string signRet = string.Empty;
using (HMACSHA1 mac = new HMACSHA1(Encoding.UTF8.GetBytes(signKey)))
{
byte[] hash = mac.ComputeHash(Encoding.UTF8.GetBytes(secret));
signRet = Convert.ToBase64String(hash);
}
return signRet;
}
public static string MakeSignPlainText(SortedDictionary<string, string> requestParams, string requestMethod, string requestHos
t, string requestPath)
{
string retStr = "";
retStr += requestMethod;
```

```
retStr += requestHost;
retStr += requestPath;
retStr += "?";
string v = "";
foreach (string key in requestParams.Keys)
{
v += string.Format("{0}={1}&", key, requestParams[key]);
}
retStr += v.TrimEnd('&');
return retStr;
}

public static void Main(string[] args)
{
// 密钥参数
string SECRET_ID = "AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******";
string SECRET_KEY = "Gu5t9xGARNpq86cd98joQYCN3*******";

string endpoint = "cvm.tencentcloudapi.com";
string region = "ap-guangzhou";
string action = "DescribeInstances";
string version = "2017-03-12";
double RequestTimestamp = 1465185768; // 时间戳 2019-02-26 00:44:25,此参数作为示例，以实际为准
// long timestamp = ToTimestamp() / 1000;
// string requestTimestamp = timestamp.ToString();
Dictionary<string, string> param = new Dictionary<string, string>();
param.Add("Limit", "20");
param.Add("Offset", "0");
param.Add("InstanceIds.0", "ins-09dx96dg");
param.Add("Action", action);
param.Add("Nonce", "11886");
// param.Add("Nonce", Math.Abs(new Random().Next()).ToString());

param.Add("Timestamp", RequestTimestamp.ToString());
param.Add("Version", version);

param.Add("SecretId", SECRET_ID);
param.Add("Region", region);
SortedDictionary<string, string> headers = new SortedDictionary<string, string>(param, StringComparer.Ordinal);
string sigInParam = MakeSignPlainText(headers, "GET", endpoint, "/");
string sigOutParam = Sign(SECRET_KEY, sigInParam);
Console.WriteLine(sigOutParam);
}
}
```

NodeJS

```
const crypto = require('crypto');
```

```
function get_req_url(params, endpoint){
params['Signature'] = escape(params['Signature']);
const url_strParam = sort_params(params)
return "https://" + endpoint + "/?" + url_strParam.slice(1);
}

function formatSignString(reqMethod, endpoint, path, strParam){
let strSign = reqMethod + endpoint + path + "?" + strParam.slice(1);
return strSign;
}
function sha1(secretKey, strsign){
let signMethodMap = {'HmacSHA1': "sha1"};
let hmac = crypto.createHmac(signMethodMap['HmacSHA1'], secretKey || "");
return hmac.update(Buffer.from(strsign, 'utf8')).digest('base64')
}

function sort_params(params){
let strParam = "";
let keys = Object.keys(params);
keys.sort();
for (let k in keys) {
//k = k.replace(/_/g, '.');
strParam += ("&" + keys[k] + "=" + params[keys[k]]);
}
return strParam
}

function main(){
// 密钥参数
const SECRET_ID = "AKIDz8krbsJ5yKBZQpn74WFkmLPx3*******"
const SECRET_KEY = "Gu5t9xGARNpq86cd98joQYCN3*******"

const endpoint = "cvm.tencentcloudapi.com"
const Region = "ap-guangzhou"
const Version = "2017-03-12"
const Action = "DescribeInstances"
const Timestamp = 1465185768 // 时间戳 2016-06-06 12:02:48, 此参数作为示例，以实际为准
// const Timestamp = Math.round(Date.now() / 1000)
const Nonce = 11886 // 随机正整数
//const nonce = Math.round(Math.random() * 65535)

let params = {};
params['Action'] = Action;
params['InstanceIds.0'] = 'ins-09dx96dg';
params['Limit'] = 20;
params['Offset'] = 0;
params['Nonce'] = Nonce;
params['Region'] = Region;
params['SecretId'] = SECRET_ID;
params['Timestamp'] = Timestamp;
```

```
params['Version'] = Version;

// 1. 对参数排序,并拼接请求字符串
strParam = sort_params(params)

// 2. 拼接签名原文字符串
const reqMethod = "GET";
const path = "/";
strSign = formatSignString(reqMethod, endpoint, path, strParam)
// console.log(strSign)

// 3. 生成签名串
params['Signature'] = sha1(SECRET_KEY, strSign)
console.log(params['Signature'])

// 4. 进行url编码并拼接请求url
// const req_url = get_req_url(params, endpoint)
// console.log(params['Signature'])
// console.log(req_url)
}
main()
```

返回结果

最近更新时间：2022-07-21 06:11:10

注意：目前只要请求被服务端正常处理了，响应的 HTTP 状态码均为200。例如返回的消息体里的错误码是签名失败，但 HTTP 状态码是200，而不是401。

正确返回结果

以云服务器的接口查看实例状态列表 (DescribeInstancesStatus) 2017-03-12 版本为例，若调用成功，其可能的返回如下为：

```
{
"Response": {
"TotalCount": 0,
"InstanceStatusSet": [],
"RequestId": "b5b41468-520d-4192-b42f-595cc34b6c1c"
}
}
```

- Response 及其内部的 RequestId 是固定的字段，无论请求成功与否，只要 API 处理了，则必定会返回。
- RequestId 用于一个 API 请求的唯一标识，如果 API 出现异常，可以联系我们，并提供该 ID 来解决问题。
- 除了固定的字段外，其余均为具体接口定义的字段，不同的接口所返回的字段参见接口文档中的定义。此例中的 TotalCount 和 InstanceStatusSet 均为 DescribeInstancesStatus 接口定义的字段，由于调用请求的用户暂时还没有云服务器实例，因此 TotalCount 在此情况下的返回值为 0， InstanceStatusSet 列表为空。

错误返回结果

若调用失败，其返回值示例如下为：

```
{
"Response": {
"Error": {
"Code": "AuthFailure.SignatureFailure",
"Message": "The provided credentials could not be validated. Please check your signature is correct."
},
"RequestId": "ed93f3cb-f35e-473f-b9f3-0d451b8b79c6"
}
}
```

- Error 的出现代表着该请求调用失败。Error 字段连同其内部的 Code 和 Message 字段在调用失败时是必定返回的。
- Code 表示具体出错的错误码，当请求出错时可以先根据该错误码在公共错误码和当前接口对应的错误码列表里面查找对应原因和解决方案。
- Message 显示出了这个错误发生的具体原因，随着业务发展或体验优化，此文本可能会经常保持变更或更新，用户不应依赖这个返回值。
- RequestId 用于一个 API 请求的唯一标识，如果 API 出现异常，可以联系我们，并提供该 ID 来解决问题。

公共错误码

返回结果中如果存在 Error 字段，则表示调用 API 接口失败。 Error 中的 Code 字段表示错误码，所有业务都可能出现的错误码为公共错误码。完整的错误码列表请参考本产品“API 文档”目录下的“错误码”页面。

参数类型

最近更新时间：2022-08-10 06:34:05

目前腾讯云 API 3.0 输入参数和输出参数支持如下几种数据格式：

- String: 字符串。
- Integer：整型，上限为无符号64位整数。SDK 3.0 不同编程语言支持的类型有所差异，建议以所使用编程语言的最大整型定义，例如 Golang 的 uint64。
- Boolean：布尔型。
- Float：浮点型。
- Double：双精度浮点型。
- Date：字符串，日期格式。例如：2022-01-01。
- Timestamp：字符串，时间格式。例如：2022-01-01 00:00:00。
- Timestamp ISO8601：ISO 8601 是由国际标准化组织（International Organization for Standardization，ISO）发布的关于日期和时间格式的国际标准，对应国标《GB/T 7408-2005数据元和交换格式信息交换日期和时间表示法》。建议以所使用编程语言的标准库进行格式解析。例如：2022-01-01T00:00:00+08:00。
- Binary：二进制内容，需要以特定协议请求和解析。

画质重生任务相关接口
创建画质重生任务

最近更新时间：2022-08-12 08:19:57

1. 接口描述

接口请求域名： ie.tencentcloudapi.com 。

创建画质重生任务，对视频进行转码、去噪、去划痕、去毛刺、超分、细节增强和色彩增强。

默认接口请求频率限制：100次/秒。

推荐使用 API Explorer　点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：CreateMediaQualityRestorationTask。
Version	是	String	公共参数，本接口取值：2020-03-04。
Region	是	String	公共参数，详见产品支持的 地域列表。
DownInfo	是	DownInfo	源文件地址。
TransInfo.N	是	Array of SubTaskTranscodeInfo	画质重生任务参数信息。
SaveInfo	是	SaveInfo	任务结束后文件存储信息。
CallbackInfo	否	CallbackInfo	任务结果回调地址信息。

3. 输出参数

参数名称	类型	描述
TaskId	String	画质重生任务ID，可以通过该ID查询任务状态。
RequestId	String	唯一请求 ID，每次请求都会返回。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 创建画质重生任务

对视频进行转码、去噪、去划痕、去毛刺、超分、细节增强和色彩增强。

输入示例

```
https://ie.tencentcloudapi.com/?Action=CreateMediaQualityRestorationTask
&DownInfo.Type=0
&DownInfo.UrlInfo.Url=http://test.video.myqcloud.com/testA.mp4
&TransInfo.0.TaskName=src_file.mp4
&TransInfo.0.TargetInfo.FileName=dst_file.mp4
&TransInfo.0.VideoInfo.VideoEnhance.ColorEnhance.Type=strong
&SaveInfo.Type=1
&SaveInfo.CosInfo.Region=ap-beijing
&SaveInfo.CosInfo.Bucket=test-123456
&SaveInfo.CosInfo.Path=/out_file/
&<公共请求参数>
```

输出示例

```
{
"Response": {
"TaskId": "b8dc83ef-7ce4-4d76-94b5-a8cfe73d40e3",
"RequestId": "5z1993x8-05y9-14a8-ab55-192ff22297c9"
}
}
```

5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python
- Tencent Cloud SDK 3.0 for Java
- Tencent Cloud SDK 3.0 for PHP
- Tencent Cloud SDK 3.0 for Go
- Tencent Cloud SDK 3.0 for NodeJS
- Tencent Cloud SDK 3.0 for .NET
- Tencent Cloud SDK 3.0 for C++
- Tencent Cloud SDK 3.0 for Ruby

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
FailedOperation.RunningTaskExceed	运行中的任务达到上限，如需要增加任务上限，请提交工单。
FailedOperation.ServerBusy	系统繁忙，请稍后重试。
FailedOperation.ServerError	内部服务错误。
InvalidParameter	参数错误。
InvalidParameterValue.CallbackUrlError	CallbackUrl 不安全。
InvalidParameterValue.TaskAlreadyDone	任务已经结束。
InvalidParameterValue.TaskDeleted	任务已经删除。
InvalidParameterValue.TaskIdNotExist	任务ID不存在。
InvalidParameterValue.UriError	请求uri错误。
InvalidParameterValue.UrlInfoUrlError	URL 不安全。
ResourceUnavailable.InArrears	帐号已欠费。
UnauthorizedOperation	未授权操作。

获取画质重生任务结果

最近更新时间：2022-08-12 08:19:56

1. 接口描述

接口请求域名： ie.tencentcloudapi.com 。

获取画质重生任务结果，查看结束后的文件信息

默认接口请求频率限制：100次/秒。

推荐使用 API Explorer

点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：DescribeMediaQualityRestorationTaskRusult。
Version	是	String	公共参数，本接口取值：2020-03-04。
Region	是	String	公共参数，详见产品支持的 地域列表。
TaskId	是	String	画质重生任务ID

3. 输出参数

参数名称	类型	描述
TaskResult	MediaQualityRestorationTaskResult	画质重生任务结果信息
RequestId	String	唯一请求 ID，每次请求都会返回。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 获取画质重生任务结果

输入示例

```
https://ie.tencentcloudapi.com/?Action=DescribeMediaQualityRestorationTaskRusult
&TaskId=b8dc83ef-7ce4-4d76-94b5-a8cfe73d40e3
&<公共请求参数>
```

输出示例

```
{
"Response": {
"TaskResult": {
"TaskId": "b8dc83ef-7ce4-4d76-94b5-a8cfe73d40e3",
"SubTaskResult": [
{
"DownloadUrl": "http://yayunhui-1251820392-1251820392.cos.ap-beijing.myqcloud.com/test/mqr_test1.mp4",
"Md5": "5adf8bd17a2239bfb1dc7172e51e34a1",
"ProgressRate": 100,
"StatusCode": 0,
"StatusMsg": "0",
"TaskName": "1080_five_all.mp4",
"FileInfo": {
"FileSize": 31254626,
"FileType": "mp4",
"Bitrate": 8323468,
"Duration": 30040,
"VideoInfoResult": [
{
"Stream": 0,
"Height": 2160,
"Bitrate": 7975062,
"Fps": "25/1",
"Codec": "h264",
"Duration": 30040,
"PixFormat": "yuv420p"
}
],
"AudioInfoResult": [
{
"Stream": 1,
"Sample": 48000,
"Channel": 6,
"Codec": "aac",
"Bitrate": 341546,
"Duration": 30015
}
]
}
}
]
},
"RequestId": "request_id_1"
}
}
```


5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python
- Tencent Cloud SDK 3.0 for Java
- Tencent Cloud SDK 3.0 for PHP
- Tencent Cloud SDK 3.0 for Go
- Tencent Cloud SDK 3.0 for NodeJS
- Tencent Cloud SDK 3.0 for .NET
- Tencent Cloud SDK 3.0 for C++
- Tencent Cloud SDK 3.0 for Ruby

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
FailedOperation.CosStorageError	Cos存储结果错误。
FailedOperation.EditError	转码时截取失败。
FailedOperation.EncodeFormatError	编码格式或参数不支持。
FailedOperation.SegmentError	转码后切片失败。
FailedOperation.ServerBusy	系统繁忙，请稍后重试。
FailedOperation.ServerError	内部服务错误。
FailedOperation.TranscodeError	转码服务异常。
FailedOperation.UnknowError	转码服务未知错误。
FailedOperation.VideoDownloadError	视频源下载失败或超时。
FailedOperation.VideoParseError	视频源解析出错。
InvalidParameter	参数错误。
InvalidParameterValue.TaskAlreadyDone	任务已经结束。
InvalidParameterValue.TaskDeleted	任务已经删除。
InvalidParameterValue.TaskIdNotExist	任务ID不存在。
InvalidParameterValue.UriError	请求uri错误。
ResourceUnavailable.InArrears	帐号已欠费。

错误码	描述
UnauthorizedOperation	未授权操作。

删除画质重生任务

最近更新时间：2022-08-12 08:19:56

1. 接口描述

接口请求域名： ie.tencentcloudapi.com 。

删除正在进行的画质重生任务

默认接口请求频率限制：100次/秒。

推荐使用 API Explorer 点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：StopMediaQualityRestorationTask。
Version	是	String	公共参数，本接口取值：2020-03-04。
Region	是	String	公共参数，详见产品支持的 地域列表。
TaskId	是	String	要删除的画质重生任务ID。

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，每次请求都会返回。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 删除画质重生任务

输入示例

```
https://ie.tencentcloudapi.com/?Action=StopMediaQualityRestorationTask
&TaskId=b8dc83ef-7ce4-4d76-94b5-a8cfe73d40e
&<公共请求参数>
```

输出示例

```
{
"Response": {
"RequestId": "request_id_1"
```

```
}
}
```

5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python
- Tencent Cloud SDK 3.0 for Java
- Tencent Cloud SDK 3.0 for PHP
- Tencent Cloud SDK 3.0 for Go
- Tencent Cloud SDK 3.0 for NodeJS
- Tencent Cloud SDK 3.0 for .NET
- Tencent Cloud SDK 3.0 for C++
- Tencent Cloud SDK 3.0 for Ruby

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
FailedOperation.ServerBusy	系统繁忙，请稍后重试。
FailedOperation.ServerError	内部服务错误。
InvalidParameter	参数错误。
InvalidParameterValue.TaskAlreadyDone	任务已经结束。
InvalidParameterValue.TaskDeleted	任务已经删除。
InvalidParameterValue.TaskIdNotExist	任务ID不存在。
InvalidParameterValue.UriError	请求uri错误。
ResourceUnavailable.InArrears	帐号已欠费。
UnauthorizedOperation	未授权操作。

媒体质检任务相关接口
创建媒体质检任务

最近更新时间：2022-08-12 08:19:59

1. 接口描述

接口请求域名： ie.tencentcloudapi.com 。

通过接口可以智能检测视频画面中抖动重影、模糊、低光照、过曝光、黑边、白边、黑屏、白屏、花屏、噪点、马赛克、二维码等在内的多个场景，还可以自动检测视频无音频异常、无声音片段。

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer　点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：CreateQualityControlTask。
Version	是	String	公共参数，本接口取值：2020-03-04。
Region	是	String	公共参数，详见产品支持的 地域列表。
QualityControlInfo	是	QualityControlInfo	质检任务参数
DownInfo	是	DownInfo	视频源信息
CallbackInfo	否	CallbackInfo	任务结果回调地址信息

3. 输出参数

参数名称	类型	描述
TaskId	String	质检任务 ID 注意：此字段可能返回 null，表示取不到有效值。
RequestId	String	唯一请求 ID，每次请求都会返回。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 创建媒体质检任务

输入示例

```
https://ie.tencentcloudapi.com/?Action=CreateQualityControlTask
&DownInfo.Type=0
```

```
&DownInfo.UrlInfo.Url=http://test.cos.ap-beijing.myqcloud.com/test.mp4
&QualityControlInfo.Interval=1000
&QualityControlInfo.Jitter=True
&QualityControlInfo.QRCode=True
&QualityControlInfo.QualityEvaluation=True
&QualityControlInfo.Voice=True
&<公共请求参数>
```

输出示例

```
{
"Response": {
"TaskId": "1260168002184429568",
"RequestId": "b228ebf6-5463-429d-8bfe-5c74fb1dc2c5"
}
}
```

5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python
- Tencent Cloud SDK 3.0 for Java
- Tencent Cloud SDK 3.0 for PHP
- Tencent Cloud SDK 3.0 for Go
- Tencent Cloud SDK 3.0 for NodeJS
- Tencent Cloud SDK 3.0 for .NET
- Tencent Cloud SDK 3.0 for C++
- Tencent Cloud SDK 3.0 for Ruby

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
FailedOperation.RunningTaskExceed	运行中的任务达到上限，如需要增加任务上限，请提交工单。

错误码	描述
FailedOperation.ServerBusy	系统繁忙，请稍后重试。
InternalError	内部错误。
InvalidParameter	参数错误。
InvalidParameterValue.CallbackUrlError	CallbackUrl 不安全。
InvalidParameterValue.CosAuthModeError	Cos授权信息错误。
InvalidParameterValue.CosHostedIdNotExist	Cos托管ID不存在。
InvalidParameterValue.DownInfoFormatWrong	视频源地址格式错误。
InvalidParameterValue.DownInfoTypeWrong	视频源下载类型错误。
InvalidParameterValue.UrlInfoUrlError	URL 不安全。
ResourceUnavailable.InArrears	帐号已欠费。
UnauthorizedOperation	未授权操作。

获取媒体质检任务结果

最近更新时间：2022-08-12 08:19:58

1. 接口描述

接口请求域名： ie.tencentcloudapi.com 。

获取媒体质检任务结果

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer 点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：DescribeQualityControlTaskResult。
Version	是	String	公共参数，本接口取值：2020-03-04。
Region	是	String	公共参数，详见产品支持的 地域列表。
TaskId	是	String	质检任务 ID

3. 输出参数

参数名称	类型	描述
TaskResult	QualityControlInfoTaskResult	质检任务结果信息
RequestId	String	唯一请求 ID，每次请求都会返回。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 查询媒体质检任务结果

输入示例

```
https://ie.tencentcloudapi.com/?Action=DescribeQualityControlTaskResult
&TaskId=1260170830789492736
&<公共请求参数>
```

输出示例

```
{
"Response": {
```

```
"TaskResult": {
"TaskId": "488022239",
"Status": 2,
"ErrCode": 0,
"ErrMsg": "",
"Progress": 100,
"UsedTime": 51,
"Duration": 150,
"QualityEvaluationScore": 74,
"JitterResults": [
{
"QualityControlItems": [
{
"StartTimeOffset": 93.92,
"EndTimeOffset": 111.16,
"Confidence": 100
},
{
"StartTimeOffset": 113.16,
"EndTimeOffset": 130.248,
"Confidence": 100
}
]
}
]
},
"RequestId": "request_id_query"
}
}
```


5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python
- Tencent Cloud SDK 3.0 for Java
- Tencent Cloud SDK 3.0 for PHP
- Tencent Cloud SDK 3.0 for Go
- Tencent Cloud SDK 3.0 for NodeJS
- Tencent Cloud SDK 3.0 for .NET
- Tencent Cloud SDK 3.0 for C++

- Tencent Cloud SDK 3.0 for Ruby

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
FailedOperation.ServerBusy	系统繁忙，请稍后重试。
FailedOperation.VideoDownloadError	视频源下载失败或超时。
FailedOperation.VideoParseError	视频源解析出错。
InternalError	内部错误。
InvalidParameter	参数错误。
InvalidParameterValue.CallbackUrlNotExist	直播场景回调地址必选。
InvalidParameterValue.CosAuthModeError	Cos授权信息错误。
InvalidParameterValue.CosHostedIdNotExist	Cos托管ID不存在。
InvalidParameterValue.DownInfoFormatWrong	视频源地址格式错误。
InvalidParameterValue.DownInfoTypeWrong	视频源下载类型错误。
InvalidParameterValue.TaskIdNotExist	任务ID不存在。
ResourceUnavailable.InArrears	帐号已欠费。
UnauthorizedOperation	未授权操作。

编辑理解相关接口
创建编辑理解任务

最近更新时间：2022-08-12 08:19:53

1. 接口描述

接口请求域名： ie.tencentcloudapi.com 。

创建编辑理解任务，可以同时选择视频标签识别、分类识别、智能拆条、智能集锦、智能封面和片头片尾识别中的一项或者多项能力。

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer

点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：CreateEditingTask。
Version	是	String	公共参数，本接口取值：2020-03-04。
Region	是	String	公共参数，详见产品支持的 地域列表。
EditingInfo	是	EditingInfo	智能编辑任务参数。
DownInfo	是	DownInfo	视频源信息。
SaveInfo	否	SaveInfo	结果存储信息。对于包含智能拆条、智能集锦或者智能封面的任务必选。
CallbackInfo	否	CallbackInfo	任务结果回调地址信息。

3. 输出参数

参数名称	类型	描述
TaskId	String	编辑任务 ID，可以通过该 ID 查询任务状态。
RequestId	String	唯一请求 ID，每次请求都会返回。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 创建智能编辑任务

对一个视频进行标签识别

输入示例

```
POST / HTTP/1.1
Host: ie.tencentcloudapi.com
Content-Type: application/json
X-TC-Action: CreateEditingTask
<公共请求参数>

{
"DownInfo": {
"UrlInfo": {
"Url": "http://test.video.myqcloud.com/testA.mp4"
},
"Type": "0"
},
"EditingInfo": {
"TagEditingInfo": {
"Switch": "1"
}
}
}
```

输出示例

```
{
"Response": {
"TaskId": "8E312055EBF7A503B92947E6D3B21EFF_1573715265075",
"RequestId": "5z1993x8-05y9-14a8-ab55-192ff22297c9"
}
}
```

5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python
- Tencent Cloud SDK 3.0 for Java
- Tencent Cloud SDK 3.0 for PHP
- Tencent Cloud SDK 3.0 for Go
- Tencent Cloud SDK 3.0 for NodeJS
- Tencent Cloud SDK 3.0 for .NET
- Tencent Cloud SDK 3.0 for C++

- Tencent Cloud SDK 3.0 for Ruby

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
FailedOperation.CosStorageError	Cos存储结果错误。
FailedOperation.RunningTaskExceed	运行中的任务达到上限，如需要增加任务上限，请提交工单。
FailedOperation.ServerBusy	系统繁忙，请稍后重试。
FailedOperation.TaskResubmit	任务重复提交。
FailedOperation.VideoDownloadError	视频源下载失败或超时。
FailedOperation.VideoParseError	视频源解析出错。
FailedOperation.VideoSizeExceed	视频大小超过限制。
InternalError	内部错误。
InvalidParameter	参数错误。
InvalidParameter.LiveStreamNotSupport	不支持直播流。
InvalidParameterValue.CallbackUrlError	CallbackUrl 不安全。
InvalidParameterValue.CallbackUrlNotExist	直播场景回调地址必选。
InvalidParameterValue.CosAuthModeError	Cos授权信息错误。
InvalidParameterValue.CosHostedIdNotExist	Cos托管ID不存在。
InvalidParameterValue.DownInfoFormatWrong	视频源地址格式错误。
InvalidParameterValue.DownInfoTypeWrong	视频源下载类型错误。
InvalidParameterValue.SaveInfoNotExist	存储信息不存在。
InvalidParameterValue.UrlInfoUrlError	URL 不安全。
InvalidParameterValue.VideoFormatError	视频格式不支持。
ResourceUnavailable.InArrears	帐号已欠费。
UnauthorizedOperation	未授权操作。

获取编辑理解任务结果

最近更新时间：2022-08-12 08:19:52

1. 接口描述

接口请求域名： ie.tencentcloudapi.com 。

获取编辑理解任务结果。

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer 点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：DescribeEditingTaskResult。
Version	是	String	公共参数，本接口取值：2020-03-04。
Region	是	String	公共参数，详见产品支持的 地域列表。
TaskId	是	String	编辑任务 ID。

3. 输出参数

参数名称	类型	描述
TaskResult	EditingTaskResult	编辑任务结果信息。
RequestId	String	唯一请求 ID，每次请求都会返回。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 获取智能编辑任务结果

输入示例

```
https://ie.tencentcloudapi.com/?Action=DescribeEditingTaskResult
&TaskId=8E312055EBF7A503B92947E6D3B21EFF_1573715265075
&<公共请求参数>
```

输出示例

```
{
"Response": {
```

```
"TaskResult": {
"TaskId": "8E312055EBF7A503B92947E6D3B21EFF_1573715265075",
"Status": 2,
"TagTaskResult": {
"Status": 2,
"ErrCode": 0,
"ErrMsg": "Success.",
"ItemSet": [
{
"Tag": "动画片",
"Confidence": 90.17
}
]
},
"ClassificationTaskResult": null,
"StripTaskResult": null,
"HighlightsTaskResult": null,
"CoverTaskResult": null,
"OpeningEndingTaskResult": null
},
"RequestId": "5d199208-24c9-44a3-ab55-192ff22207c9"
}
}
```


5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python
- Tencent Cloud SDK 3.0 for Java
- Tencent Cloud SDK 3.0 for PHP
- Tencent Cloud SDK 3.0 for Go
- Tencent Cloud SDK 3.0 for NodeJS
- Tencent Cloud SDK 3.0 for .NET
- Tencent Cloud SDK 3.0 for C++
- Tencent Cloud SDK 3.0 for Ruby

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
InternalError	内部错误。
InvalidParameter	参数错误。
InvalidParameterValue.TaskIdNotExist	任务ID不存在。
ResourceUnavailable.InArrears	帐号已欠费。
UnauthorizedOperation	未授权操作。

编辑处理相关接口
创建编辑处理任务

最近更新时间：2022-08-12 08:19:55

1. 接口描述

接口请求域名： ie.tencentcloudapi.com 。

用于创建编辑处理任务，如媒体截取、媒体编辑、媒体拼接、媒体字幕。

默认接口请求频率限制：50次/秒。

推荐使用 API Explorer　点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：CreateMediaProcessTask。
Version	是	String	公共参数，本接口取值：2020-03-04。
Region	是	String	公共参数，详见产品支持的 地域列表。
MediaProcessInfo	是	MediaProcessInfo	编辑处理任务参数。
SourceInfoSet.N	否	Array of MediaSourceInfo	编辑处理任务输入源列表。
SaveInfoSet.N	否	Array of SaveInfo	结果存储信息，对于涉及存储的请求必选。部子任务支持数组备份写，具体以对应任务文档为准。
CallbackInfoSet.N	否	Array of CallbackInfo	任务结果回调地址信息。部子任务支持数组备份回调，具体以对应任务文档为准。

3. 输出参数

参数名称	类型	描述
TaskId	String	编辑任务 ID，可以通过该 ID 查询任务状态和结果。 注意：此字段可能返回 null，表示取不到有效值。
RequestId	String	唯一请求 ID，每次请求都会返回。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 创建编辑处理/剪切任务

输入示例

```
POST / HTTP/1.1
Host: ie.tencentcloudapi.com
Content-Type: application/json
X-TC-Action: CreateMediaProcessTask
<公共请求参数>

{
"CallbackInfoSet": [
{
"Url": "http://qq.com"
}
],
"SaveInfoSet": [
{
"CosInfo": {
"Path": "/video_cutting/test_res/1012_test_sec_video",
"Region": "ap-beijing",
"Bucket": "shortvideo-1251820392",
"CosAuthMode": {
"HostedId": "internal_test",
"Type": "1"
}
},
"Type": "1"
}
],
"MediaProcessInfo": {
"MediaCuttingInfo": {
"TargetInfo": {
"Format": "jpg",
"TargetVideoInfo": {
"Width": "800",
"Height": "800"
},
"FileName": "dstfile-{index|:5|4:x}"
},
"ResultListSaveType": "UseSaveInfo",
"TimeInfo": {
"IntervalPoint": {
"Interval": "5000",
"StartTime": "20000"
},
"Type": "IntervalPoint"
},
"OutForm": {
"FillType": "Gaussian",
"Type": "Static"
}
},
```

```
"Type": "MediaCutting"
},
"SourceInfoSet": [
{
"DownInfo": {
"CosInfo": {
"Path": "/video/sar_dar/sar_dar.mov",
"Region": "ap-beijing",
"Bucket": "shortvideo-1251820392",
"CosAuthMode": {
"HostedId": "internal_test",
"Type": "1"
}
},
"Type": "1"
},
"Type": "Video",
"Id": "source_video"
}
]
}
```

输出示例

```
{
"Response": {
"TaskId": "1200132608361473955430 4",
"RequestId": "5z1993x8-05y9-14a8-ab55-192ff22297c9"
}
}
```

5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python
- Tencent Cloud SDK 3.0 for Java
- Tencent Cloud SDK 3.0 for PHP
- Tencent Cloud SDK 3.0 for Go
- Tencent Cloud SDK 3.0 for NodeJS

- Tencent Cloud SDK 3.0 for .NET
- Tencent Cloud SDK 3.0 for C++
- Tencent Cloud SDK 3.0 for Ruby

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
AuthFailure	CAM签名/鉴权错误。
FailedOperation	操作失败。
FailedOperation.RunningTaskExceed	运行中的任务达到上限，如需要增加任务上限，请提交工单。
FailedOperation.ServerBusy	系统繁忙，请稍后重试。
FailedOperation.ServerError	内部服务错误。
FailedOperation.VideoDownloadError	视频源下载失败或超时。
FailedOperation.VideoParseError	视频源解析出错。
InternalError	内部错误。
InvalidParameter	参数错误。
InvalidParameterValue	参数取值错误。
InvalidParameterValue.CallbackUrlError	CallbackUrl 不安全。
InvalidParameterValue.DownInfoFormatWrong	视频源地址格式错误。
InvalidParameterValue.DownInfoTypeWrong	视频源下载类型错误。
InvalidParameterValue.LiveSourceNotSupport	该任务不支持直播流。
InvalidParameterValue.TaskIdNotExist	任务ID不存在。
InvalidParameterValue.UrlInfoUrlError	URL 不安全。
InvalidParameterValue.VideoFormatError	视频格式不支持。
RequestLimitExceeded	请求的次数超过了频率限制。
ResourceUnavailable.InArrears	帐号已欠费。

获取编辑处理任务结果

最近更新时间：2022-08-12 08:19:55

1. 接口描述

接口请求域名： ie.tencentcloudapi.com 。

用于获取编辑处理任务的结果。

默认接口请求频率限制：50次/秒。

推荐使用 API Explorer 点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：DescribeMediaProcessTaskResult。
Version	是	String	公共参数，本接口取值：2020-03-04。
Region	是	String	公共参数，详见产品支持的 地域列表。
TaskId	是	String	编辑处理任务ID。

3. 输出参数

参数名称	类型	描述
TaskResult	MediaProcessTaskResult	任务处理结果。 注意：此字段可能返回 null，表示取不到有效值。
RequestId	String	唯一请求 ID，每次请求都会返回。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 查询编辑处理任务结果

输入示例

```
POST / HTTP/1.1
Host: ie.tencentcloudapi.com
Content-Type: application/json
X-TC-Action: DescribeMediaProcessTaskResult
<公共请求参数>

{
```

```
"TaskId": "1200132608361473955430 4"
}
```

输出示例

```
{
"Response": {
"TaskResult": {
"ErrCode": 100,
"ErrMsg": "Task is under processing.",
"MediaCuttingTaskResult": null,
"Progress": 24,
"Status": 1200,
"TaskId": "1200133178704604144435 2",
"Type": "MediaCutting"
},
"RequestId": "e4de90c2-2836-44d4-8513-f4ff06596fe9"
}
}
```

5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python
- Tencent Cloud SDK 3.0 for Java
- Tencent Cloud SDK 3.0 for PHP
- Tencent Cloud SDK 3.0 for Go
- Tencent Cloud SDK 3.0 for NodeJS
- Tencent Cloud SDK 3.0 for .NET
- Tencent Cloud SDK 3.0 for C++
- Tencent Cloud SDK 3.0 for Ruby

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
InvalidParameterValue.TaskIdNotExist	任务ID不存在。

停止编辑处理任务

最近更新时间：2022-08-12 08:19:54

1. 接口描述

接口请求域名： ie.tencentcloudapi.com 。

用于停止正在进行中的编辑处理任务。

默认接口请求频率限制：50次/秒。

推荐使用 API Explorer 点击调试

API Explorer 提供了在线调用、签名验证、SDK 代码生成和快速检索接口等能力。您可查看每次调用的请求内容和返回结果以及自动生成 SDK 调用示例。

2. 输入参数

以下请求参数列表仅列出了接口请求参数和部分公共参数，完整公共参数列表见 公共请求参数。

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：StopMediaProcessTask。
Version	是	String	公共参数，本接口取值：2020-03-04。
Region	是	String	公共参数，详见产品支持的 地域列表。
TaskId	是	String	编辑处理任务ID。

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，每次请求都会返回。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 停止编辑处理/剪切任务

输入示例

```
POST / HTTP/1.1
Host: ie.tencentcloudapi.com
Content-Type: application/json
X-TC-Action: StopMediaProcessTask
<公共请求参数>

{
"TaskId": "1200132608361473955430 4"
}
```

输出示例

```
{
"Response": {
"RequestId": "5d199208-24c9-44a3-ab55-192ff22207c9"
}
}
```

5. 开发者资源

腾讯云 API 平台

腾讯云 API 平台 是综合 API 文档、错误码、API Explorer 及 SDK 等资源的统一查询平台，方便您从同一入口查询及使用腾讯云提供的所有 API 服务。

API Inspector

用户可通过 API Inspector 查看控制台每一步操作关联的 API 调用情况，并自动生成各语言版本的 API 代码，也可前往 API Explorer 进行在线调试。

SDK

云 API 3.0 提供了配套的开发工具集（SDK），支持多种编程语言，能更方便的调用 API。

- Tencent Cloud SDK 3.0 for Python
- Tencent Cloud SDK 3.0 for Java
- Tencent Cloud SDK 3.0 for PHP
- Tencent Cloud SDK 3.0 for Go
- Tencent Cloud SDK 3.0 for NodeJS
- Tencent Cloud SDK 3.0 for .NET
- Tencent Cloud SDK 3.0 for C++
- Tencent Cloud SDK 3.0 for Ruby

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
InternalError	内部错误。
InvalidParameter	参数错误。
InvalidParameterValue.ActionNotSupport	任务不支持该操作。
InvalidParameterValue.TaskIdNotExist	任务ID不存在。
ResourceUnavailable.InArrears	帐号已欠费。

数据结构

最近更新时间：2022-07-05 06:14:28

ArtifactReduction

去编码毛刺、伪影参数

被如下接口引用：CreateMediaQualityRestorationTask。

名称	类型	必选	描述
Type	String	否	去毛刺方式：weak,,strong
Algorithm	String	否	去毛刺算法，可选项： edaf， wdaf， 默认edaf。 注意：此参数已经弃用

AudioEnhance

音频音效增强，只支持无背景音的音频

被如下接口引用：CreateMediaQualityRestorationTask。

名称	类型	必选	描述
Type	String	否	音效增强种类，可选项：normal

AudioInfo

音频参数信息

被如下接口引用：CreateMediaQualityRestorationTask。

名称	类型	必选	描述
Bitrate	Integer	是	音频码率，取值范围：0 和 [26, 256]，单位：kbps。 注意：当取值为 0，表示音频码率和原始音频保持一致。
Codec	String	是	音频编码器，可选项：aac,mp3,ac3,flac,mp2。
Channel	Integer	是	声道数，可选项： 1：单声道， 2：双声道， 6：立体声。
SampleRate	Integer	是	采样率，单位：Hz。可选项：32000，44100,48000
Denoise	Denoise	否	音频降噪信息
EnableMuteAudio	Integer	否	开启添加静音，可选项： 0：不开启， 1：开启， 默认不开启
LoudnessInfo	LoudnessInfo	否	音频响度信息
AudioEnhance	AudioEnhance	否	音频音效增强

名称	类型	必选	描述
RemoveReverb	RemoveReverb	否	去除混音

AudioInfoResultItem

任务结束后生成的文件音频信息

被如下接口引用：DescribeMediaQualityRestorationTaskRusult。

名称	类型	描述
Stream	Integer	音频流的流id。
Sample	Integer	音频采样率 。 注意：此字段可能返回 null，表示取不到有效值。
Channel	Integer	音频声道数。 注意：此字段可能返回 null，表示取不到有效值。
Codec	String	编码格式，如aac, mp3等。 注意：此字段可能返回 null，表示取不到有效值。
Bitrate	Integer	码率，单位：bps。 注意：此字段可能返回 null，表示取不到有效值。
Duration	Integer	音频时长，单位：ms。 注意：此字段可能返回 null，表示取不到有效值。

CallbackInfo

任务结果回调地址信息

被如下接口引用：CreateEditingTask, CreateMediaProcessTask, CreateMediaQualityRestorationTask, CreateQualityControlTask。

名称	类型	必选	描述
Url	String	是	回调URL。

ClassificationEditingInfo

视频分类识别任务参数信息

被如下接口引用：CreateEditingTask。

名称	类型	必选	描述
Switch	Integer	是	是否开启视频分类识别。0为关闭，1为开启。其他非0非1值默认为0。
CustomInfo	String	否	额外定制化服务参数。参数为序列化的Json字符串，例如：{"k1":"v1"}。

ClassificationTaskResult

视频分类识别结果信息

被如下接口引用：DescribeEditingTaskResult。

名称	类型	描述

名称	类型	描述
Status	Integer	编辑任务状态。 1：执行中；2：成功；3：失败。
ErrCode	Integer	编辑任务失败错误码。 0：成功；其他值：失败。
ErrMsg	String	编辑任务失败错误描述。
ItemSet	Array of ClassificationTaskResultItem	视频分类识别结果集。 注意：此字段可能返回 null，表示取不到有效值。

ClassificationTaskResultItem

视频分类识别结果项

被如下接口引用：DescribeEditingTaskResult。

名称	类型	描述
Classification	String	分类名称。
Confidence	Float	置信度，取值范围是 0 到 100。

ColorEnhance

颜色增强参数

被如下接口引用：CreateMediaQualityRestorationTask。

名称	类型	必选	描述
Type	String	是	颜色增强类型，可选项： 1. tra； 2. weak； 3. normal； 4. strong； 注意：tra不支持自适应调整，处理速度更快；weak,normal,strong支持基于画面颜色自适应，处理速度更慢。

CosAuthMode

任务视频cos授权信息

被如下接口引用：CreateEditingTask, CreateMediaProcessTask, CreateMediaQualityRestorationTask, CreateQualityControlTask。

名称	类型	必选	描述
Type	Integer	是	授权类型，可选值： 0：bucket授权，需要将对应bucket授权给本服务帐号（3020447271和100012301793），否则会读写cos失败； 1：key托管，把cos的账号id和key托管于本服务，本服务会提供一个托管id； 3：临时key授权。 注意：目前智能编辑还不支持临时key授权；画质重生目前只支持bucket授权
HostedId	String	否	cos账号托管id，Type等于1时必选。
SecretId	String	否	cos身份识别id，Type等于3时必选。

名称	类型	必选	描述
SecretKey	String	否	cos身份秘钥，Type等于3时必选。
Token	String	否	临时授权 token，Type等于3时必选。

CosInfo

任务视频cos信息

被如下接口引用：CreateEditingTask, CreateMediaProcessTask, CreateMediaQualityRestorationTask, CreateQualityControlTask。

名称	类型	必选	描述
Region	String	是	cos 区域值。例如：ap-beijing。
Bucket	String	是	cos 存储桶，格式为BuketName-AppId。例如：test-123456。
Path	String	是	cos 路径。 对于写表示目录，例如：/test； 对于读表示文件路径，例如：/test/test.mp4。
CosAuthMode	CosAuthMode	否	cos 授权信息，不填默认为公有权限。

CoverEditingInfo

智能封面任务参数信息

被如下接口引用：CreateEditingTask。

名称	类型	必选	描述
Switch	Integer	是	是否开启智能封面。0为关闭，1为开启。其他非0非1值默认为0。
CustomInfo	String	否	额外定制化服务参数。参数为序列化的Json字符串，例如：{"k1":"v1"}。

CoverTaskResult

智能封面结果信息

被如下接口引用：DescribeEditingTaskResult。

名称	类型	描述
Status	Integer	编辑任务状态。 1：执行中；2：成功；3：失败。
ErrCode	Integer	编辑任务失败错误码。 0：成功；其他值：失败。
ErrMsg	String	编辑任务失败错误描述。
ItemSet	Array of CoverTaskResultItem	智能封面结果集。 注意：此字段可能返回 null，表示取不到有效值。

CoverTaskResultItem

智能封面结果项

被如下接口引用：DescribeEditingTaskResult。

名称	类型	描述
CoverUrl	String	智能封面地址。
Confidence	Float	置信度，取值范围是 0 到 100。

DarInfo

视频Dar信息

被如下接口引用：CreateMediaQualityRestorationTask。

名称	类型	必选	描述
FillMode	Integer	否	填充模式，可选值： 1：留黑，保持视频宽高比不变，边缘剩余部分使用黑色填充。 2：拉伸，对每一帧进行拉伸，填满整个画面，可能导致转码后的视频被“压扁“或者“拉长“。 默认为2。

Denoise

音频降噪

被如下接口引用：CreateMediaQualityRestorationTask。

名称	类型	必选	描述
Type	String	否	音频降噪强度，可选项： 1. weak 2.normal， 3.strong 默认为weak

Denoising

去噪参数

被如下接口引用：CreateMediaQualityRestorationTask。

名称	类型	必选	描述
Type	String	是	去噪方式，可选项： templ：时域降噪； spatial：空域降噪， fast-spatial：快速空域降噪。 注意：可选择组合方式： 1.type:"templ,spatial"； 2.type:"templ,fast-spatial"。
TemplStrength	Float	否	时域去噪强度，可选值：0.0-1.0 。小于0.0的默认为0.0，大于1.0的默认为1.0。
SpatialStrength	Float	否	空域去噪强度，可选值：0.0-1.0 。小于0.0的默认为0.0，大于1.0的默认为1.0。

DownInfo

视频源信息

被如下接口引用：CreateEditingTask, CreateMediaProcessTask, CreateMediaQualityRestorationTask, CreateQualityControlTask。

名称	类型	必选	描述
Type	Integer	是	下载类型，可选值： 0：UrlInfo； 1：CosInfo。
UrlInfo	UrlInfo	否	Url形式下载信息，当Type等于0时必选。
CosInfo	CosInfo	否	Cos形式下载信息，当Type等于1时必选。

DynamicImageInfo

动图参数

被如下接口引用：CreateMediaProcessTask。

名称	类型	必选	描述
Quality	Integer	否	画面质量，范围：1~100。 • 对于webp格式，默认：75 • 对于gif格式，小于10为低质量，大于50为高质量，其它为普通。默认：低质量。

EditInfo

画质重生子任务视频剪辑参数

被如下接口引用：CreateMediaQualityRestorationTask。

名称	类型	必选	描述
StartTime	Integer	是	剪辑开始时间，单位：ms。
EndTime	Integer	是	剪辑结束时间，单位：ms

EditingInfo

智能编辑任务参数信息

被如下接口引用：CreateEditingTask。

名称	类型	必选	描述
TagEditingInfo	TagEditingInfo	否	视频标签识别任务参数，不填则不开启。
ClassificationEditingInfo	ClassificationEditingInfo	否	视频分类识别任务参数，不填则不开启。
StripEditingInfo	StripEditingInfo	否	智能拆条任务参数，不填则不开启。
HighlightsEditingInfo	HighlightsEditingInfo	否	智能集锦任务参数，不填则不开启。
CoverEditingInfo	CoverEditingInfo	否	智能封面任务参数，不填则不开启。
OpeningEndingEditingInfo	OpeningEndingEditingInfo	否	片头片尾识别任务参数，不填则不开启。

EditingTaskResult

智能识别任务结果信息

被如下接口引用：DescribeEditingTaskResult。

名称	类型	描述
TaskId	String	编辑任务 ID。
Status	Integer	编辑任务状态。 1：执行中；2：已完成。
TagTaskResult	TagTaskResult	视频标签识别结果。 注意：此字段可能返回 null，表示取不到有效值。
ClassificationTaskResult	ClassificationTaskResult	视频分类识别结果。 注意：此字段可能返回 null，表示取不到有效值。
StripTaskResult	StripTaskResult	智能拆条结果。 注意：此字段可能返回 null，表示取不到有效值。
HighlightsTaskResult	HighlightsTaskResult	智能集锦结果。 注意：此字段可能返回 null，表示取不到有效值。
CoverTaskResult	CoverTaskResult	智能封面结果。 注意：此字段可能返回 null，表示取不到有效值。
OpeningEndingTaskResult	OpeningEndingTaskResult	片头片尾识别结果。 注意：此字段可能返回 null，表示取不到有效值。

FaceProtect

人脸保护参数

被如下接口引用：CreateMediaQualityRestorationTask。

名称	类型	必选	描述
FaceUsmRatio	Float	否	人脸区域增强强度，可选项：0.0-1.0。小于0.0的默认为0.0，大于1.0的默认为1.0。

FileInfo

画质重生处理后文件的详细信息

被如下接口引用：DescribeMediaQualityRestorationTaskRusult。

名称	类型	描述
FileSize	Integer	任务结束后生成的文件大小。 注意：此字段可能返回 null，表示取不到有效值。
FileType	String	任务结束后生成的文件格式，例如：mp4,flv等等。 注意：此字段可能返回 null，表示取不到有效值。
Bitrate	Integer	任务结束后生成的文件整体码率，单位：bps。 注意：此字段可能返回 null，表示取不到有效值。
Duration	Integer	任务结束后生成的文件时长，单位：ms。 注意：此字段可能返回 null，表示取不到有效值。
VideoInfoResult	Array of VideoInfoResultItem	任务结束后生成的文件视频信息。 注意：此字段可能返回 null，表示取不到有效值。

名称	类型	描述
AudioInfoResult	Array of AudioInfoResultItem	任务结束后生成的文件音频信息。 注意：此字段可能返回 null，表示取不到有效值。

FrameTagItem

帧标签

被如下接口引用：DescribeMediaProcessTaskResult。

名称	类型	描述
StartPts	Integer	标签起始时间戳PTS(ms)
EndPts	Integer	语句结束时间戳PTS(ms)
Period	String	字符串形式的起始结束时间
TagItems	Array of TagItem	标签数组

FrameTagRec

帧标签任务参数

被如下接口引用：CreateMediaProcessTask。

名称	类型	必选	描述
TagType	String	是	标签类型： "Common": 通用类型 "Game":游戏类型
GameExtendType	String	否	游戏具体类型： "HonorOfKings_AnchorViews":王者荣耀主播视角 "HonorOfKings_GameViews":王者荣耀比赛视角 "LOL_AnchorViews":英雄联盟主播视角 "LOL_GameViews":英雄联盟比赛视角 "PUBG_AnchorViews":和平精英主播视角 "PUBG_GameViews":和平精英比赛视角

FrameTagResult

帧标签结果

被如下接口引用：DescribeMediaProcessTaskResult。

名称	类型	描述
FrameTagItems	Array of FrameTagItem	帧标签结果数组

HiddenMarkInfo

数字水印

被如下接口引用：CreateMediaQualityRestorationTask。

名称	类型	必选	描述

名称	类型	必选	描述
Path	String	是	数字水印路径,，如果不从Cos拉取水印，则必填
Frequency	Integer	否	数字水印频率，可选值：[1,256]，默认值为30
Strength	Integer	否	数字水印强度，可选值：[32,128]，默认值为64
CosInfo	CosInfo	否	数字水印的Cos 信息，从Cos上拉取图片水印时必填。

HighlightsEditingInfo

智能集锦任务参数信息

被如下接口引用：CreateEditingTask。

名称	类型	必选	描述
Switch	Integer	是	是否开启智能集锦。0为关闭，1为开启。其他非0非1值默认为0。
CustomInfo	String	否	额外定制化服务参数。参数为序列化的Json字符串，例如：{"k1":"v1"}。

HighlightsTaskResult

智能集锦结果信息

被如下接口引用：DescribeEditingTaskResult。

名称	类型	描述
Status	Integer	编辑任务状态。 1：执行中；2：成功；3：失败。
ErrCode	Integer	编辑任务失败错误码。 0：成功；其他值：失败。
ErrMsg	String	编辑任务失败错误描述。
ItemSet	Array of HighlightsTaskResultItem	智能集锦结果集。 注意：此字段可能返回 null，表示取不到有效值。

HighlightsTaskResultItem

智能集锦结果项

被如下接口引用：DescribeEditingTaskResult。

名称	类型	描述
HighlightUrl	String	智能集锦地址。
CovImgUrl	String	智能集锦封面地址。
Confidence	Float	置信度，取值范围是 0 到 100。
Duration	Float	智能集锦持续时间，单位：秒。
SegmentSet	Array of HighlightsTaskResultItemSegment	智能集锦子片段结果集，集锦片段由这些子片段拼接生成。

HighlightsTaskResultItemSegment

智能集锦结果片段

被如下接口引用：DescribeEditingTaskResult。

名称	类型	描述
Confidence	Float	置信度，取值范围是 0 到 100。
StartTimeOffset	Float	集锦片段起始的偏移时间，单位：秒。
EndTimeOffset	Float	集锦片段终止的偏移时间，单位：秒。

IntervalTime

周期时间点信息。

被如下接口引用：CreateMediaProcessTask。

名称	类型	必选	描述
Interval	Integer	是	间隔周期，单位ms
StartTime	Integer	否	开始时间点，单位ms

LoudnessInfo

音频响度信息

被如下接口引用：CreateMediaQualityRestorationTask。

名称	类型	必选	描述
Loudness	Float	否	音频整体响度
LoudnessRange	Float	否	音频响度范围

LowLightEnhance

低光照增强参数

被如下接口引用：CreateMediaQualityRestorationTask。

名称	类型	必选	描述
Type	String	是	低光照增强类型，可选项：normal。

MediaCuttingInfo

编辑处理/剪切任务信息。
截图结果默认存在 SaveInfoSet 的第一个存储位置。

被如下接口引用：CreateMediaProcessTask。

名称	类型	必选	描述
TimeInfo	MediaCuttingTimeInfo	是	截取时间信息。
TargetInfo	MediaTargetInfo	是	输出结果信息。

名称	类型	必选	描述
OutForm	MediaCuttingOutForm	是	截取结果形式信息。
ResultListSaveType	String	否	列表文件形式，存储到用户存储服务中，可选值： • NoListFile：不存储结果列表； • UseSaveInfo：默认，结果列表和结果存储同一位置（即 SaveInfoSet 的第一个存储位置）； • SaveInfoSet 存储的Id：存储在指定的存储位置。
WatermarkInfoSet	Array of MediaCuttingWatermark	否	水印信息，最多支持 10 个水印。
DropPureColor	String	否	是否去除纯色截图，如果值为 True ，对应时间点的截图如果是纯色，将略过。

MediaCuttingOutForm

编辑处理/剪切任务/输出形式信息

被如下接口引用：CreateMediaProcessTask。

名称	类型	必选	描述
Type	String	是	输出类型，可选值： Static：静态图； Dynamic：动态图； Sprite：雪碧图； Video：视频。 注1：不同类型时，对应的 TargetInfo.Format 格式支持如下： Static：jpg、png； Dynamic：gif； Sprite：jpg、png； Video：mp4。 注2：当 Type=Sprite时，TargetInfo指定的尺寸表示小图的大小，最终结果尺寸以输出为准。
FillType	String	否	背景填充方式，可选值： White：白色填充； Black：黑色填充； Stretch：拉伸； Gaussian：高斯模糊； 默认White。
SpriteRowCount	Integer	否	【废弃】参考SpriteInfo
SpriteColumnCount	Integer	否	【废弃】参考SpriteInfo
SpriteInfo	SpriteImageInfo	否	Type=Sprite时有效，表示雪碧图参数信息。
DynamicInfo	DynamicImageInfo	否	Type=Dynamic时有效，表示动图参数信息。

MediaCuttingTaskResult

编辑处理/剪切任务/处理结果

被如下接口引用：DescribeMediaProcessTaskResult。

名称	类型	描述
ListFile	TaskResultFile	如果ResultListType不为NoListFile时，结果（TaskResultFile）列表文件的存储位置。 注意：此字段可能返回 null，表示取不到有效值。
ResultCount	Integer	结果个数。 注意：此字段可能返回 null，表示取不到有效值。
FirstFile	TaskResultFile	第一个结果文件。 注意：此字段可能返回 null，表示取不到有效值。
LastFile	TaskResultFile	最后一个结果文件。 注意：此字段可能返回 null，表示取不到有效值。
ImageCount	Integer	任务结果包含的图片总数。 静态图：总数即为文件数； 雪碧图：所有小图总数； 动图、视频：不计算图片数，为 0。 注意：此字段可能返回 null，表示取不到有效值。

MediaCuttingTimeInfo

编辑处理/剪切任务/时间信息

被如下接口引用：CreateMediaProcessTask。

名称	类型	必选	描述
Type	String	是	时间类型，可选值： PointSet：时间点集合； IntervalPoint：周期采样点； SectionSet：时间片段集合。
PointSet	Array of Integer	否	截取时间点集合，单位毫秒，Type=PointSet时必选。
IntervalPoint	IntervalTime	否	周期采样点信息，Type=IntervalPoint时必选。
SectionSet	Array of SectionTime	否	时间区间集合信息，Type=SectionSet时必选。

MediaCuttingWatermark

媒体剪切水印信息。

被如下接口引用：CreateMediaProcessTask。

名称	类型	必选	描述
Type	String	是	水印类型，可选值： • Image：图像水印； • Text：文字水印。
Image	MediaCuttingWatermarkImage	否	图像水印信息，当 Type=Image 时必选。
Text	MediaCuttingWatermarkText	否	文字水印信息，当 Type=Text 时必选。

MediaCuttingWatermarkImage

媒体剪切图像水印参数。

被如下接口引用：CreateMediaProcessTask。

名称	类型	必选	描述
SourceId	String	是	水印源的ID，对应SourceInfoSet内的源。 注意1：对应的 MediaSourceInfo.Type需要为Image。 注意2：对于动图，只取第一帧图像作为水印源。
PosX	Integer	否	水印水平坐标，单位像素，默认：0。
PosY	Integer	否	水印垂直坐标，单位像素，默认：0。
Width	Integer	否	水印宽度，单位像素，默认：0。
Height	Integer	否	水印高度，单位像素，默认：0。 注意：对于宽高符合以下规则： 1、Width>0 且 Height>0，按指定宽高拉伸； 2、Width=0 且 Height>0，以Height为基准等比缩放； 3、Width>0 且 Height=0，以Width为基准等比缩放； 4、Width=0 且 Height=0，采用源的宽高。
PosOriginType	String	否	指定坐标原点，可选值： • LeftTop：PosXY 表示水印左上点到图片左上点的相对位置 • RightTop：PosXY 表示水印右上点到图片右上点的相对位置 • LeftBottom：PosXY 表示水印左下点到图片左下点的相对位置 • RightBottom：PosXY 表示水印右下点到图片右下点的相对位置 • Center：PosXY 表示水印中心点到图片中心点的相对位置 默认：LeftTop。

MediaCuttingWatermarkText

媒体剪切文字水印参数。

被如下接口引用：CreateMediaProcessTask。

名称	类型	必选	描述
Text	String	是	水印文字。
FontSize	Integer	是	文字大小
PosX	Integer	否	水印水平坐标，单位像素，默认：0。
PosY	Integer	否	水印垂直坐标，单位像素，默认：0。
FontColor	String	否	文字颜色，格式为：#RRGGBBAA，默认值：#000000。
FontAlpha	Integer	否	文字透明度，范围：0~100，默认值：100。
PosOriginType	String	否	指定坐标原点，可选值： • LeftTop：PosXY 表示水印左上点到图片左上点的相对位置 • RightTop：PosXY 表示水印右上点到图片右上点的相对位置 • LeftBottom：PosXY 表示水印左下点到图片左下点的相对位置 • RightBottom：PosXY 表示水印右下点到图片右下点的相对位置 • Center：PosXY 表示水印中心点到图片中心点的相对位置 默认：LeftTop。

名称	类型	必选	描述
Font	String	否	字体，可选值： • SimHei • SimKai • Arial 默认 SimHei。

MediaJoiningInfo

编辑处理/拼接任务信息

被如下接口引用：CreateMediaProcessTask。

名称	类型	必选	描述
TargetInfo	MediaTargetInfo	是	输出目标信息，拼接只采用FileName和Format，用于指定目标文件名和格式。 其中Format只支持mp4.
Mode	String	否	拼接模式： Fast：快速； Normal：正常；

MediaJoiningTaskResult

编辑处理/拼接任务/处理结果

被如下接口引用：DescribeMediaProcessTaskResult。

名称	类型	描述
File	TaskResultFile	拼接结果文件。 注意：此字段可能返回 null，表示取不到有效值。

MediaProcessInfo

编辑处理/任务信息

被如下接口引用：CreateMediaProcessTask。

名称	类型	必选	描述
Type	String	是	编辑处理任务类型，可选值： MediaEditing：媒体编辑（待上线）； MediaCutting：媒体剪切； MediaJoining：媒体拼接。 MediaRecognition: 媒体识别。
MediaCuttingInfo	MediaCuttingInfo	否	视频剪切任务参数，Type=MediaCutting时必选。
MediaJoiningInfo	MediaJoiningInfo	否	视频拼接任务参数，Type=MediaJoining时必选。
MediaRecognitionInfo	MediaRecognitionInfo	否	媒体识别任务参数，Type=MediaRecognition时必选

MediaProcessTaskResult

编辑处理/任务处理结果

被如下接口引用：DescribeMediaProcessTaskResult。

名称	类型	描述
TaskId	String	编辑处理任务ID。 注意：此字段可能返回 null，表示取不到有效值。
Type	String	编辑处理任务类型，取值： MediaEditing：视频编辑（待上线）； MediaCutting：视频剪切； MediaJoining：视频拼接。 MediaRecognition：媒体识别； 注意：此字段可能返回 null，表示取不到有效值。
Progress	Integer	处理进度，范围：[0,100] 注意：此字段可能返回 null，表示取不到有效值。
Status	Integer	任务状态： 1100：等待中； 1200：执行中； 2000：成功； 5000：失败。 注意：此字段可能返回 null，表示取不到有效值。
ErrCode	Integer	任务错误码。 注意：此字段可能返回 null，表示取不到有效值。
ErrMsg	String	任务错误信息。 注意：此字段可能返回 null，表示取不到有效值。
MediaCuttingTaskResult	MediaCuttingTaskResult	剪切任务处理结果，当Type=MediaCutting时才有效。 注意：此字段可能返回 null，表示取不到有效值。
MediaJoiningTaskResult	MediaJoiningTaskResult	拼接任务处理结果，当Type=MediaJoining时才有效。 注意：此字段可能返回 null，表示取不到有效值。
MediaRecognitionTaskResult	MediaRecognitionTaskResult	媒体识别任务处理结果，当Type=MediaRecognition时才有效。 注意：此字段可能返回 null，表示取不到有效值。

MediaQualityRestorationTaskResult

画质重生任务结果

被如下接口引用：DescribeMediaQualityRestorationTaskRusult。

名称	类型	描述
TaskId	String	画质重生任务ID
SubTaskResult	Array of SubTaskResultItem	画质重生处理后文件的详细信息。

MediaRecognitionInfo

媒体识别任务参数

被如下接口引用：CreateMediaProcessTask。

名称	类型	必选	描述
FrameTagRec	FrameTagRec	否	帧标签识别

名称	类型	必选	描述
SubtitleRec	SubtitleRec	否	语音字幕识别

MediaRecognitionTaskResult

媒体识别任务处理结果

被如下接口引用：DescribeMediaProcessTaskResult。

名称	类型	描述
FrameTagResults	FrameTagResult	帧标签识别结果 注意：此字段可能返回 null，表示取不到有效值。
SubtitleResults	SubtitleResult	语音字幕识别结果 注意：此字段可能返回 null，表示取不到有效值。

MediaResultInfo

结果文件媒体信息

被如下接口引用：DescribeMediaProcessTaskResult。

名称	类型	描述
Duration	Integer	媒体时长，单位：毫秒 注意：此字段可能返回 null，表示取不到有效值。
ResultVideoInfoSet	Array of ResultVideoInfo	视频流信息 注意：此字段可能返回 null，表示取不到有效值。
ResultAudioInfoSet	Array of ResultAudioInfo	音频流信息 注意：此字段可能返回 null，表示取不到有效值。

MediaSourceInfo

编辑处理的媒体源

被如下接口引用：CreateMediaProcessTask。

名称	类型	必选	描述
DownInfo	DownInfo	是	媒体源资源下载信息。
Id	String	否	媒体源ID标记，用于多个输入源时，请内媒体源的定位，对于多输入的任务，一般要求必选。 ID只能包含字母、数字、下划线、中划线，长读不能超过128。
Type	String	否	媒体源类型，具体类型如下： Video：视频 Image：图片 Audio：音频

MediaTargetInfo

目标媒体信息。

被如下接口引用：CreateMediaProcessTask。

名称	类型	必选	描述
FileName	String	是	目标文件名，不能带特殊字符（如/等），无需后缀名，最长200字符。 注1：部分子服务支持占位符，形式为：{parameter} 预设parameter有： index：序号；
Format	String	是	媒体封装格式，最长5字符，具体格式支持根据子任务确定。
TargetVideoInfo	TargetVideoInfo	否	视频流信息。
ResultListSaveType	String	否	【不再使用】

MuxInfo

流封装信息

被如下接口引用：CreateMediaQualityRestorationTask。

名称	类型	必选	描述
DeleteStream	String	否	删除流，可选项：video,audio。
FlvFlags	String	否	Flv 参数，目前支持add_keyframe_index

OpeningEndingEditingInfo

片头片尾识别任务参数信息

被如下接口引用：CreateEditingTask。

名称	类型	必选	描述
Switch	Integer	是	是否开启片头片尾识别。0为关闭，1为开启。其他非0非1值默认为0。
CustomInfo	String	否	额外定制化服务参数。参数为序列化的Json字符串，例如：{"k1":"v1"}。

OpeningEndingTaskResult

片头片尾识别结果信息

被如下接口引用：DescribeEditingTaskResult。

名称	类型	描述
Status	Integer	编辑任务状态。 1：执行中；2：成功；3：失败。
ErrCode	Integer	编辑任务失败错误码。 0：成功；其他值：失败。
ErrMsg	String	编辑任务失败错误描述。
Item	OpeningEndingTaskResultItem	片头片尾识别结果项。 注意：此字段可能返回 null，表示取不到有效值。

OpeningEndingTaskResultItem

片头片尾识别结果项

被如下接口引用：DescribeEditingTaskResult。

名称	类型	描述
OpeningTimeOffset	Float	视频片头的结束时间点，单位：秒。
OpeningConfidence	Float	片头识别置信度，取值范围是 0 到 100。
EndingTimeOffset	Float	视频片尾的开始时间点，单位：秒。
EndingConfidence	Float	片尾识别置信度，取值范围是 0 到 100。

PicMarkInfoItem

图片水印信息

被如下接口引用：CreateMediaQualityRestorationTask。

名称	类型	必选	描述
PosX	Integer	是	图片水印的X坐标。
PosY	Integer	是	图片水印的Y坐标 。
Path	String	否	图片水印路径,，如果不从Cos拉取水印，则必填
CosInfo	CosInfo	否	图片水印的Cos 信息，从Cos上拉取图片水印时必填。
Width	Integer	否	图片水印宽度，不填为图片原始宽度。
Height	Integer	否	图片水印高度，不填为图片原始高度。
StartTime	Integer	否	添加图片水印的开始时间,单位：ms。
EndTime	Integer	否	添加图片水印的结束时间,单位：ms。

QualityControlInfo

媒体质检任务参数信息

被如下接口引用：CreateQualityControlTask。

名称	类型	必选	描述
Interval	Integer	否	对流进行截图的间隔ms，默认1000ms
VideoShot	Boolean	否	是否保存截图
Jitter	Boolean	否	是否检测抖动重影
Blur	Boolean	否	是否检测模糊
AbnormalLighting	Boolean	否	是否检测低光照、过曝
CrashScreen	Boolean	否	是否检测花屏
BlackWhiteEdge	Boolean	否	是否检测黑边、白边、黑屏、白屏、绿屏
Noise	Boolean	否	是否检测噪点
Mosaic	Boolean	否	是否检测马赛克

名称	类型	必选	描述
QRCode	Boolean	否	是否检测二维码，包括小程序码、条形码
QualityEvaluation	Boolean	否	是否开启画面质量评价
QualityEvalScore	Integer	否	画面质量评价过滤阈值，结果只返回低于阈值的时间段，默认60
Voice	Boolean	否	是否检测视频音频，包含静音、低音、爆音

QualityControlInfoTaskResult

媒体质检结果信息

被如下接口引用：DescribeQualityControlTaskResult。

名称	类型	描述
TaskId	String	质检任务 ID
Status	Integer	质检任务状态。 1：执行中；2：成功；3：失败
Progress	Integer	表示处理进度百分比
UsedTime	Integer	处理时长(s)
Duration	Integer	计费时长(s)
NoAudio	Boolean	为true时表示视频无音频轨 注意：此字段可能返回 null，表示取不到有效值。
NoVideo	Boolean	为true时表示视频无视频轨 注意：此字段可能返回 null，表示取不到有效值。
QualityEvaluationScore	Integer	视频无参考质量打分，百分制 注意：此字段可能返回 null，表示取不到有效值。
QualityEvaluationResults	Array of QualityControlResultItems	视频画面无参考评分低于阈值的时间段 注意：此字段可能返回 null，表示取不到有效值。
JitterResults	Array of QualityControlResultItems	视频画面抖动时间段 注意：此字段可能返回 null，表示取不到有效值。
BlurResults	Array of QualityControlResultItems	视频画面模糊时间段 注意：此字段可能返回 null，表示取不到有效值。
AbnormalLightingResults	Array of QualityControlResultItems	视频画面低光、过曝时间段 注意：此字段可能返回 null，表示取不到有效值。
CrashScreenResults	Array of QualityControlResultItems	视频画面花屏时间段 注意：此字段可能返回 null，表示取不到有效值。
BlackWhiteEdgeResults	Array of QualityControlResultItems	视频画面黑边、白边、黑屏、白屏、纯色屏时间段 注意：此字段可能返回 null，表示取不到有效值。
NoiseResults	Array of QualityControlResultItems	视频画面有噪点时间段 注意：此字段可能返回 null，表示取不到有效值。
MosaicResults	Array of QualityControlResultItems	视频画面有马赛克时间段 注意：此字段可能返回 null，表示取不到有效值。

名称	类型	描述
QRCodeResults	Array of QualityControlResultItems	视频画面有二维码的时间段，包括小程序码、条形码 注意：此字段可能返回 null，表示取不到有效值。
VoiceResults	Array of QualityControlResultItems	视频音频异常时间段，包括静音、低音、爆音 注意：此字段可能返回 null，表示取不到有效值。
ErrCode	Integer	任务错误码 注意：此字段可能返回 null，表示取不到有效值。
ErrMsg	String	任务错误信息 注意：此字段可能返回 null，表示取不到有效值。

QualityControlItem

质检结果项

被如下接口引用：DescribeQualityControlTaskResult。

名称	类型	描述
Confidence	Integer	置信度，取值范围是 0 到 100 注意：此字段可能返回 null，表示取不到有效值。
StartTimeOffset	Float	出现的起始时间戳，秒
EndTimeOffset	Float	出现的结束时间戳，秒
AreaCoordsSet	Array of Integer	区域坐标(px)，即左上角坐标、右下角坐标 注意：此字段可能返回 null，表示取不到有效值。

QualityControlResultItems

质检结果项数组

被如下接口引用：DescribeQualityControlTaskResult。

名称	类型	描述
Id	String	异常类型 注意：此字段可能返回 null，表示取不到有效值。
QualityControlItems	Array of QualityControlItem	质检结果项

RemoveReverb

音频去除混响

被如下接口引用：CreateMediaQualityRestorationTask。

名称	类型	必选	描述
Type	String	否	去混响类型，可选项：normal

ResultAudioInfo

结果媒体文件的视频流信息

被如下接口引用：DescribeMediaProcessTaskResult。

名称	类型	描述
StreamId	Integer	流在媒体文件中的流ID 注意：此字段可能返回 null，表示取不到有效值。
Duration	Integer	流的时长，单位：毫秒 注意：此字段可能返回 null，表示取不到有效值。

ResultVideoInfo

结果媒体文件的视频流信息

被如下接口引用：DescribeMediaProcessTaskResult。

名称	类型	描述
StreamId	Integer	流在媒体文件中的流ID 注意：此字段可能返回 null，表示取不到有效值。
Duration	Integer	流的时长，单位：毫秒 注意：此字段可能返回 null，表示取不到有效值。
Width	Integer	画面宽度 注意：此字段可能返回 null，表示取不到有效值。
Height	Integer	画面高度 注意：此字段可能返回 null，表示取不到有效值。
Fps	Integer	视频帧率，如果高于原始帧率，部分服务将无效。 注意：此字段可能返回 null，表示取不到有效值。

SaveInfo

任务存储信息

被如下接口引用：CreateEditingTask, CreateMediaProcessTask, CreateMediaQualityRestorationTask。

名称	类型	必选	描述
Type	Integer	是	存储类型，可选值： 1：CosInfo。
CosInfo	CosInfo	否	Cos形式存储信息，当Type等于1时必选。
Id	String	否	存储信息ID标记，用于多个输出场景。部分任务支持多输出时，一般要求必选。 ID只能包含字母、数字、下划线、中划线，长读不能超过128。

ScratchRepair

去划痕参数

被如下接口引用：CreateMediaQualityRestorationTask。

名称	类型	必选	描述
Type	String	否	去划痕方式，取值：normal。
Ratio	Float	否	去划痕强度， 可选项：0.0-1.0。小于0.0的默认为0.0，大于1.0的默认为1.0。

SectionTime

时间区间。

被如下接口引用：CreateMediaProcessTask。

名称	类型	必选	描述
StartTime	Integer	是	开始时间点，单位ms
Duration	Integer	是	时间区间时长，单位ms

SegmentInfo

输出文件切片信息

被如下接口引用：CreateMediaQualityRestorationTask。

名称	类型	必选	描述
FragmentTime	Integer	否	每个切片平均时长，默认10s。
SegmentType	String	否	切片类型，可选项：hls，不填时默认hls。
FragmentName	String	否	切片文件名字。注意： 1.不填切片文件名时，默认按照按照如下格式命名：m3u8文件名{order}。 2.若填了切片文件名字，则会按照如下格式命名：用户指定文件名{order}。

Sharp

细节增强参数

被如下接口引用：CreateMediaQualityRestorationTask。

名称	类型	必选	描述
Type	String	否	细节增强方式,取值：normal。
Ratio	Float	否	细节增强强度，可选项：0.0-1.0。小于0.0的默认为0.0，大于1.0的默认为1.0。

SpriteImageInfo

雪碧图参数信息

注意：雪碧图大图整体的宽和高都不能大于 65000 像素。

被如下接口引用：CreateMediaProcessTask。

名称	类型	必选	描述
RowCount	Integer	否	表示雪碧图行数，默认：10。
ColumnCount	Integer	否	表示雪碧图列数，默认：10。
MarginTop	Integer	否	第一行元素与顶部像素距离，默认：0。
MarginBottom	Integer	否	最后一行元素与底部像素距离，默认：0。
MarginLeft	Integer	否	最左一行元素与左边像素距离，默认：0。
MarginRight	Integer	否	最右一行元素与右边像素距离，默认：0。

名称	类型	必选	描述
PaddingTop	Integer	否	小图与元素顶部像素距离，默认：0。
PaddingBottom	Integer	否	小图与元素底部像素距离，默认：0。
PaddingLeft	Integer	否	小图与元素左边像素距离，默认：0。
PaddingRight	Integer	否	小图与元素右边像素距离，默认：0。
BackgroundColor	String	否	背景颜色，格式：#RRGGBB，默认：#FFFFFF。

StripEditingInfo

智能拆条任务参数信息

被如下接口引用：CreateEditingTask。

名称	类型	必选	描述
Switch	Integer	是	是否开启智能拆条。0为关闭，1为开启。其他非0非1值默认为0。
CustomInfo	String	否	额外定制化服务参数。参数为序列化的Json字符串，例如：{"k1":"v1"}。

StripTaskResult

智能拆条结果信息

被如下接口引用：DescribeEditingTaskResult。

名称	类型	描述
Status	Integer	编辑任务状态。 1：执行中；2：成功；3：失败。
ErrCode	Integer	编辑任务失败错误码。 0：成功；其他值：失败。
ErrMsg	String	编辑任务失败错误描述。
ItemSet	Array of StripTaskResultItem	智能拆条结果集。 注意：此字段可能返回 null，表示取不到有效值。

StripTaskResultItem

智能拆条结果项

被如下接口引用：DescribeEditingTaskResult。

名称	类型	描述
SegmentUrl	String	视频拆条片段地址。
CovImgUrl	String	拆条封面图片地址。
Confidence	Float	置信度，取值范围是 0 到 100。
StartTimeOffset	Float	拆条片段起始的偏移时间，单位：秒。
EndTimeOffset	Float	拆条片段终止的偏移时间，单位：秒。

SubTaskResultItem

画质重生子任务结果

被如下接口引用：DescribeMediaQualityRestorationTaskRusult。

名称	类型	描述
TaskName	String	子任务名称。 注意：此字段可能返回 null，表示取不到有效值。
StatusCode	Integer	子任务状态。 0：成功； 1：执行中； 其他值：失败。
StatusMsg	String	子任务状态描述。
ProgressRate	Integer	子任务进度。 注意：此字段可能返回 null，表示取不到有效值。
DownloadUrl	String	画质重生处理后文件的下载地址。 注意：此字段可能返回 null，表示取不到有效值。
Md5	String	画质重生处理后文件的MD5。 注意：此字段可能返回 null，表示取不到有效值。
FileInfo	FileInfo	画质重生处理后文件的详细信息。 注意：此字段可能返回 null，表示取不到有效值。

SubTaskTranscodeInfo

画质重生子任务参数信息

被如下接口引用：CreateMediaQualityRestorationTask。

名称	类型	必选	描述
TaskName	String	是	子任务名称。
TargetInfo	TargetInfo	是	目标文件信息。
EditInfo	EditInfo	否	视频剪辑信息。注意：如果填写了EditInfo，则VideoInfo和AudioInfo必填
VideoInfo	VideoInfo	否	视频转码信息，不填保持和源文件一致。
AudioInfo	AudioInfo	否	音频转码信息，不填保持和源文件一致。
MuxInfo	MuxInfo	否	指定封装信息。

SubtitleItem

语音字幕识别项

被如下接口引用：DescribeMediaProcessTaskResult。

名称	类型	描述
Id	String	语音识别结果
Zh	String	中文翻译结果 注意：此字段可能返回 null，表示取不到有效值。

名称	类型	描述
En	String	英文翻译结果 注意：此字段可能返回 null，表示取不到有效值。
StartPts	Integer	语句起始时间戳PTS(ms)
EndPts	Integer	语句结束时间戳PTS(ms)
Period	String	字符串形式的起始结束时间
Confidence	Integer	结果的置信度（百分制）
EndFlag	Boolean	当前语句是否结束
PuncEndTs	String	语句分割时间戳 注意：此字段可能返回 null，表示取不到有效值。

SubtitleRec

语音字幕任务参数

被如下接口引用：CreateMediaProcessTask。

名称	类型	必选	描述
AsrDst	String	否	语音识别： zh：中文 en：英文
TransDst	String	否	翻译识别： zh：中文 en：英文

SubtitleResult

语音字幕识别结果

被如下接口引用：DescribeMediaProcessTaskResult。

名称	类型	描述
SubtitleItems	Array of SubtitleItem	语音字幕数组

TagEditingInfo

视频标签识别任务参数信息

被如下接口引用：CreateEditingTask。

名称	类型	必选	描述
Switch	Integer	是	是否开启视频标签识别。0为关闭，1为开启。其他非0非1值默认为0。
CustomInfo	String	否	额外定制化服务参数。参数为序列化的Json字符串，例如：{"k1":"v1"}。

TagItem

标签项

被如下接口引用：DescribeMediaProcessTaskResult。

名称	类型	描述
Id	String	标签内容
Confidence	Integer	结果的置信度（百分制）
Categorys	Array of String	分级数组 注意：此字段可能返回 null，表示取不到有效值。
Ext	String	标签备注 注意：此字段可能返回 null，表示取不到有效值。

TagTaskResult

视频标签识别结果信息

被如下接口引用：DescribeEditingTaskResult。

名称	类型	描述
Status	Integer	编辑任务状态。 1：执行中；2：成功；3：失败。
ErrCode	Integer	编辑任务失败错误码。 0：成功；其他值：失败。
ErrMsg	String	编辑任务失败错误描述。
ItemSet	Array of TagTaskResultItem	视频标签识别结果集。 注意：此字段可能返回 null，表示取不到有效值。

TagTaskResultItem

视频标签识别结果项

被如下接口引用：DescribeEditingTaskResult。

名称	类型	描述
Tag	String	标签名称。
Confidence	Float	置信度，取值范围是 0 到 100。

TargetInfo

输出文件信息

被如下接口引用：CreateMediaQualityRestorationTask。

名称	类型	必选	描述
FileName	String	是	目标文件名
SegmentInfo	SegmentInfo	否	目标文件切片信息

TargetVideoInfo

目标视频信息。

被如下接口引用：CreateMediaProcessTask。

名称	类型	必选	描述
Width	Integer	否	视频宽度，单位像素，一般要求是偶数，否则会向下对齐。
Height	Integer	否	视频高度，单位像素，一般要求是偶数，否则会向下对齐。
FrameRate	Integer	否	视频帧率，范围在1到120之间

TaskResultFile

任务结果文件信息

被如下接口引用：DescribeMediaProcessTaskResult。

名称	类型	描述
Url	String	文件链接。 注意：此字段可能返回 null，表示取不到有效值。
FileSize	Integer	文件大小，部分任务支持，单位：字节 注意：此字段可能返回 null，表示取不到有效值。
MediaInfo	MediaResultInfo	媒体信息，对于媒体文件，部分任务支持返回 注意：此字段可能返回 null，表示取不到有效值。
Md5	String	文件对应的md5。 注意：此字段可能返回 null，表示取不到有效值。

TextMarkInfoItem

画质重生子任务文字水印信息

被如下接口引用：CreateMediaQualityRestorationTask。

名称	类型	必选	描述
Text	String	是	文字内容。
PosX	Integer	是	文字水印X坐标。
PosY	Integer	是	文字水印Y坐标。
FontSize	Integer	是	文字大小
FontFile	String	否	字体，可选项：hei,song，simkai,arial；默认hei(黑体）。
FontColor	String	否	字体颜色，颜色见附录，不填默认black。
FontAlpha	Float	否	文字透明度，可选值0-1。0：不透明，1：全透明。默认为0

UrlInfo

任务视频Url形式下载信息。

被如下接口引用：CreateEditingTask, CreateMediaQualityRestorationTask, CreateQualityControlTask。

名称	类型	必选	描述

名称	类型	必选	描述
Url	String	是	视频 URL。 注意：编辑理解仅支持mp4、flv等格式的点播文件，不支持hls；
Format	Integer	否	视频地址格式，可选值： 0：音视频； 1：直播流。 默认为0。其他非0非1值默认为0。画质重生任务只支持0。
Host	String	否	【不再支持】指定请求资源时，HTTP头部host的值。

VideoEnhance

画质增强参数信息

被如下接口引用：CreateMediaQualityRestorationTask。

名称	类型	必选	描述
ArtifactReduction	ArtifactReduction	否	去编码毛刺、伪影参数。
Denoising	Denoising	否	去噪声参数。
ColorEnhance	ColorEnhance	否	颜色增强参数。
Sharp	Sharp	否	细节增强参数。
WdSuperResolution	Integer	否	超分参数，可选项：2，目前仅支持2倍超分。 注意：此参数已经弃用，超分可以使用VideoSuperResolution参数
FaceProtect	FaceProtect	否	人脸保护信息。
WdFps	Integer	否	插帧，取值范围：[0, 60]，单位：Hz。 注意：当取值为 0，表示帧率和原始视频保持一致。
ScratchRepair	ScratchRepair	否	去划痕参数
LowLightEnhance	LowLightEnhance	否	低光照增强参数
VideoSuperResolution	VideoSuperResolution	否	视频超分参数
VideoRepair	VideoRepair	否	视频画质修复参数

VideoInfo

视频转码信息

被如下接口引用：CreateMediaQualityRestorationTask。

名称	类型	必选	描述
Fps	Integer	否	视频帧率，取值范围：[0, 60]，单位：Hz。 注意：当取值为 0，表示帧率和原始视频保持一致。
Width	Integer	否	宽度，取值范围：0 和 [128, 4096] 注意： 当 Width、Height 均为 0，则分辨率同源； 当 Width 为 0，Height 非 0，则 Width 按比例缩放； 当 Width 非 0，Height 为 0，则 Height 按比例缩放； 当 Width、Height 均非 0，则分辨率按用户指定。

名称	类型	必选	描述
Height	Integer	否	高度，取值范围：0 和 [128, 4096] 注意： 当 Width、Height 均为 0，则分辨率同源； 当 Width 为 0，Height 非 0，则 Width 按比例缩放； 当 Width 非 0，Height 为 0，则 Height 按比例缩放； 当 Width、Height 均非 0，则分辨率按用户指定。
LongSide	Integer	否	长边分辨率，取值范围：0 和 [128, 4096] 注意： 当 LongSide、ShortSide 均为 0，则分辨率按照Width，Height； 当 LongSide 为 0，ShortSide 非 0，则 LongSide 按比例缩放； 当 LongSide非 0，ShortSide为 0，则 ShortSide 按比例缩放； 当 LongSide、ShortSide 均非 0，则分辨率按用户指定。 长短边优先级高于Weight,Height,设置长短边则忽略宽高。
ShortSide	Integer	否	短边分辨率，取值范围：0 和 [128, 4096] 注意： 当 LongSide、ShortSide 均为 0，则分辨率按照Width，Height； 当 LongSide 为 0，ShortSide 非 0，则 LongSide 按比例缩放； 当 LongSide非 0，ShortSide为 0，则 ShortSide 按比例缩放； 当 LongSide、ShortSide 均非 0，则分辨率按用户指定。 长短边优先级高于Weight,Height,设置长短边则忽略宽高。
Bitrate	Integer	否	视频流的码率，取值范围：0 和 [128, 35000]，单位：kbps。当取值为 0，表示视频码率和原始视频保持一致。
Gop	Integer	否	固定I帧之间，视频帧数量，取值范围： [25, 2500]，如果不填，使用编码默认最优序列。
VideoCodec	String	否	编码器支持选项，可选值： h264， h265， av1。 不填默认h264。
PicMarkInfo	Array of PicMarkInfoItem	否	图片水印。
DarInfo	DarInfo	否	填充方式，当视频流配置宽高参数与原始视频的宽高比不一致时，对转码的处理方式，即为“填充”。
Hdr	String	否	支持hdr,可选项： hdr10， hlg。 此时，VideoCodec会强制设置为h265, 编码位深为10
VideoEnhance	VideoEnhance	否	画质增强参数信息。
HiddenMarkInfo	HiddenMarkInfo	否	数字水印参数信息。
TextMarkInfo	Array of TextMarkInfoItem	否	文本水印参数信息。

VideoInfoResultItem

任务结束后生成的文件视频信息

被如下接口引用：DescribeMediaQualityRestorationTaskRusult。

名称	类型	描述
Stream	Integer	视频流的流id。
Width	Integer	视频宽度。 注意：此字段可能返回 null，表示取不到有效值。
Height	Integer	视频高度。 注意：此字段可能返回 null，表示取不到有效值。
Bitrate	Integer	视频码率，单位：bps。 注意：此字段可能返回 null，表示取不到有效值。
Fps	String	视频帧率，用分数格式表示，如：25/1, 99/32等等。 注意：此字段可能返回 null，表示取不到有效值。
Codec	String	编码格式，如h264,h265等等 。 注意：此字段可能返回 null，表示取不到有效值。
Rotate	Integer	播放旋转角度，可选值0-360。 注意：此字段可能返回 null，表示取不到有效值。
Duration	Integer	视频时长，单位：ms 。 注意：此字段可能返回 null，表示取不到有效值。
PixFormat	String	颜色空间，如yuv420p，yuv444p等等。 注意：此字段可能返回 null，表示取不到有效值。

VideoRepair

综合画质修复，包括：去噪，去毛刺，细节增强，主观画质提升。

被如下接口引用：CreateMediaQualityRestorationTask。

名称	类型	必选	描述
Type	String	否	画质修复类型，可选值：weak，normal，strong; 默认值: weak

VideoSuperResolution

视频超分

被如下接口引用：CreateMediaQualityRestorationTask。

名称	类型	必选	描述
Type	String	否	超分视频类型：可选值：lq,hq lq: 针对低清晰度有较多噪声视频的超分; hq: 针对高清晰度视频超分; 默认取值：lq。
Size	Integer	否	超分倍数，可选值：2。 注意：当前只支持两倍超分。

错误码

最近更新时间：2022-07-15 06:10:26

功能说明

如果返回结果中存在 Error 字段，则表示调用 API 接口失败。例如：

```
{
"Response": {
"Error": {
"Code": "AuthFailure.SignatureFailure",
"Message": "The provided credentials could not be validated. Please check your signature is correct."
},
"RequestId": "ed93f3cb-f35e-473f-b9f3-0d451b8b79c6"
}
}
```

Error 中的 Code 表示错误码，Message 表示该错误的具体信息。

错误码列表

公共错误码

错误码	说明
ActionOffline	接口已下线。
AuthFailure.InvalidAuthorization	请求头部的 Authorization 不符合腾讯云标准。
AuthFailure.InvalidSecretId	密钥非法（不是云 API 密钥类型）。
AuthFailure.MFAFailure	MFA 错误。
AuthFailure.SecretIdNotFound	密钥不存在。请在 控制台 检查密钥是否已被删除或者禁用，如状态正常，请检查密钥是否填写正确，注意前后不得有空格。
AuthFailure.SignatureExpire	签名过期。Timestamp 和服务器时间相差不得超过五分钟，请检查本地时间是否和标准时间同步。
AuthFailure.SignatureFailure	签名错误。签名计算错误，请对照调用方式中的签名方法文档检查签名计算过程。
AuthFailure.TokenFailure	token 错误。
AuthFailure.UnauthorizedOperation	请求未授权。请参考 CAM 文档对鉴权的说明。
DryRunOperation	DryRun 操作，代表请求将会是成功的，只是多传了 DryRun 参数。
FailedOperation	操作失败。
InternalError	内部错误。
InvalidAction	接口不存在。
InvalidParameter	参数错误（包括参数格式、类型等错误）。
InvalidParameterValue	参数取值错误。

错误码	说明
InvalidRequest	请求 body 的 multipart 格式错误。
IpInBlacklist	IP地址在黑名单中。
IpNotInWhitelist	IP地址不在白名单中。
LimitExceeded	超过配额限制。
MissingParameter	缺少参数。
NoSuchProduct	产品不存在
NoSuchVersion	接口版本不存在。
RequestLimitExceeded	请求的次数超过了频率限制。
RequestLimitExceeded.GlobalRegionUinLimitExceeded	主账号超过频率限制。
RequestLimitExceeded.IPLimitExceeded	IP限频。
RequestLimitExceeded.UinLimitExceeded	主账号限频。
RequestSizeLimitExceeded	请求包超过限制大小。
ResourceInUse	资源被占用。
ResourceInsufficient	资源不足。
ResourceNotFound	资源不存在。
ResourceUnavailable	资源不可用。
ResponseSizeLimitExceeded	返回包超过限制大小。
ServiceUnavailable	当前服务暂时不可用。
UnauthorizedOperation	未授权操作。
UnknownParameter	未知参数错误，用户多传未定义的参数会导致错误。
UnsupportedOperation	操作不支持。
UnsupportedProtocol	http(s) 请求协议错误，只支持 GET 和 POST 请求。
UnsupportedRegion	接口不支持所传地域。

业务错误码

错误码	说明
AuthFailure	CAM签名/鉴权错误。
FailedOperation.CosStorageError	Cos存储结果错误。
FailedOperation.EditError	转码时截取失败。
FailedOperation.EncodeFormatError	编码格式或参数不支持。
FailedOperation.RunningTaskExceed	运行中的任务达到上限，如需要增加任务上限，请提交工单。
FailedOperation.SegmentError	转码后切片失败。
FailedOperation.ServerBusy	系统繁忙，请稍后重试。

错误码	说明
FailedOperation.ServerError	内部服务错误。
FailedOperation.TaskResubmit	任务重复提交。
FailedOperation.TranscodeError	转码服务异常。
FailedOperation.UnknowError	转码服务未知错误。
FailedOperation.VideoDownloadError	视频源下载失败或超时。
FailedOperation.VideoParseError	视频源解析出错。
FailedOperation.VideoSizeExceed	视频大小超过限制。
InvalidParameter.LiveStreamNotSupport	不支持直播流。
InvalidParameterValue.ActionNotSupport	任务不支持该操作。
InvalidParameterValue.CallbackUrlError	CallbackUrl 不安全。
InvalidParameterValue.CallbackUrlNotExist	直播场景回调地址必选。
InvalidParameterValue.CosAuthModeError	Cos授权信息错误。
InvalidParameterValue.CosHostedIdNotExist	Cos托管ID不存在。
InvalidParameterValue.DownInfoFormatWrong	视频源地址格式错误。
InvalidParameterValue.DownInfoTypeWrong	视频源下载类型错误。
InvalidParameterValue.LiveSourceNotSupport	该任务不支持直播流。
InvalidParameterValue.SaveInfoNotExist	存储信息不存在。
InvalidParameterValue.TaskAlreadyDone	任务已经结束。
InvalidParameterValue.TaskDeleted	任务已经删除。
InvalidParameterValue.TaskIdNotExist	任务ID不存在。
InvalidParameterValue.UriError	请求uri错误。
InvalidParameterValue.UrlInfoUrlError	URL 不安全。
InvalidParameterValue.VideoFormatError	视频格式不支持。
ResourceUnavailable.InArrears	帐号已欠费。