

消息队列 RocketMQ 版 API 文档 (5.x)



腾讯云

【版权声明】

©2013-2024 腾讯云版权所有

本文档（含所有文字、数据、图片等内容）完整的著作权归腾讯云计算（北京）有限责任公司单独所有，未经腾讯云事先明确书面许可，任何主体不得以任何形式复制、修改、使用、抄袭、传播本文档全部或部分的内容。前述行为构成对腾讯云著作权的侵犯，腾讯云将依法采取措施追究法律责任。

【商标声明】

及其它腾讯云服务相关的商标均为腾讯云计算（北京）有限责任公司及其关联公司所有。本文档涉及的第三方主体的商标，依法由权利人所有。未经腾讯云及有关权利人书面许可，任何主体不得以任何方式对前述商标进行使用、复制、修改、传播、抄录等行为，否则将构成对腾讯云及有关权利人商标权的侵犯，腾讯云将依法采取措施追究法律责任。

【服务声明】

本文档意在向您介绍腾讯云全部或部分产品、服务的当时的相关概况，部分产品、服务的内容可能不时有所调整。

您所购买的腾讯云产品、服务的种类、服务标准等应由您与腾讯云之间的商业合同约定，除非双方另有约定，否则，腾讯云对本文档内容不做任何明示或默示的承诺或保证。

【联系我们】

我们致力于为您提供个性化的售前购买咨询服务，及相应的技术售后服务，任何问题请联系 4009100100或95716。

文档目录

API 文档 (5.x)

更新历史

简介

API 概览

调用方式

请求结构

公共参数

签名方法 v3

签名方法

返回结果

参数类型

主题相关接口

创建主题

删除主题

修改主题属性

查询主题列表

查询主题详情

消费组相关接口

创建消费组

删除消费组

修改消费组属性

查询消费组详情

查询消费组列表

角色鉴权相关接口

添加角色

查询角色列表

修改角色

删除角色

平滑迁移相关接口

平滑迁移：导入消费者组列表

平滑迁移：导入topic列表

MQTT相关接口

获取MQTT实例列表

查询MQTT实例详情信息

创建MQTT主题

删除MQTT主题

修改MQTT主题属性

查询MQTT主题列表

查询MQTT消息列表

添加MQTT角色

查询MQTT用户列表

修改MQTT角色

删除MQTT角色

获取MQTT产品售卖规格

购买MQTT实例

删除MQTT实例

修改MQTT实例属性

查询MQTT主题详情

- 更新MQTT集群证书证书
- 查询MQTT集群证书列表
- 查询 MQTT 客户端详情
- 为MQTT实例创建公网接入点
- 删除MQTT实例的公网接入点
- 查询MQTT实例公网接入点
- 查询MQTT实例VPC接入点
- 更新MQTT实例公网接入点
- 查询MQTT消息

集群相关接口

- 购买实例
- 删除实例
- 修改实例属性
- 获取实例列表
- 查询实例信息

数据结构

- 错误码

API 文档 (5.x)

更新历史

最近更新时间: 2024-04-22 01:28:25

第 14 次发布

发布时间: 2024-04-22 01:28:17

本次发布包含了以下内容:

改善已有的文档。

修改接口:

- [DescribeMQTTClient](#)
 - 新增出参: Inbound, OutBound, CleanSession
- [DescribeMQTTInsPublicEndpoints](#)
 - 新增出参: Status

新增数据结构:

- [PacketStatistics](#)
- [StatisticsReport](#)

修改数据结构:

- [MQTTEndpointItem](#)
 - 新增成员: Ip

第 13 次发布

发布时间: 2024-04-18 01:31:47

本次发布包含了以下内容:

改善已有的文档。

删除接口:

- [DescribeTopicStatsOp](#)

第 12 次发布

发布时间: 2024-04-04 01:25:24

本次发布包含了以下内容:

改善已有的文档。

修改接口:

- [ModifyTopic](#)
 - 新增入参: MsgTTL

第 11 次发布

发布时间: 2024-03-07 01:26:39

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- [CreateInstance](#)
 - 新增入参：PayMode, RenewFlag, TimeSpan, MaxTopicNum
- [CreateTopic](#)
 - 新增入参：MsgTTL
- [DescribeTopic](#)
 - 修改入参：Offset, Limit
 - 新增出参：MsgTTL
- [DescribeTopicList](#)
 - 修改入参：Offset, Limit
- [ModifyInstance](#)
 - 新增入参：MaxTopicNum

修改数据结构：

- [SubscriptionData](#)
 - 新增成员：MessageModel
 - 修改成员：InstanceId, Topic, TopicType, TopicQueueNum, ConsumerGroup, IsOnline, ConsumeType, SubString, ExpressionType, Consistency, ConsumerLag, LastUpdateTime, MaxRetryTimes, ConsumeMessageOrderly
- [TopicItem](#)
 - 新增成员：ClusterIdV4, NamespaceV4, TopicV4, FullNamespaceV4, MsgTTL

第 10 次发布

发布时间：2024-03-06 01:26:32

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- [DescribeInstance](#)
 - 新增出参：ScaledTpsEnabled, RenewFlag, ExpiryTime, RoleNumLimit, AclEnabled, TopicNumLowerLimit, TopicNumUpperLimit

修改数据结构：

- [Endpoint](#)
 - 修改成员：Type, Status, PayMode, EndpointUrl, VpId, SubnetId, Bandwidth, IpRules

第 9 次发布

发布时间：2024-01-23 01:30:19

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- [CreateMQTTInsPublicEndpoint](#)
 - 修改入参：Rules

第 8 次发布

发布时间：2024-01-22 01:23:11

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- [CreateMQTTInsPublicEndpoint](#)
- [CreateMQTTTopic](#)
- [CreateMQTTUser](#)
- [DeleteMQTTInsPublicEndpoint](#)
- [DeleteMQTTInstance](#)
- [DeleteMQTTTopic](#)
- [DeleteMQTTUser](#)
- [DescribeMQTTClient](#)
- [DescribeMQTTInsPublicEndpoints](#)
- [DescribeMQTTInsVPCEndpoints](#)
- [DescribeMQTTInstance](#)
- [DescribeMQTTInstanceCert](#)
- [DescribeMQTTMessage](#)
- [DescribeMQTTMessageList](#)
- [DescribeMQTTTopic](#)
- [DescribeMQTTTopicList](#)
- [DescribeMQTTUserList](#)
- [ModifyMQTTInsPublicEndpoint](#)
- [ModifyMQTTInstance](#)
- [ModifyMQTTInstanceCertBinding](#)
- [ModifyMQTTTopic](#)
- [ModifyMQTTUser](#)

新增数据结构：

- [CustomMapEntry](#)
- [MQTTClientSubscription](#)
- [MQTTEndpointItem](#)
- [MQTTMessageItem](#)
- [MQTTTopicItem](#)
- [MQTTUserItem](#)
- [PublicAccessRule](#)

第 7 次发布

发布时间：2024-01-19 01:23:15

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- [CreateMQTTInstance](#)
- [DescribeMQTTInstanceList](#)
- [DescribeMQTTProductSKUList](#)

新增数据结构:

- [MQTTInstanceItem](#)
- [MQTTProductSkulItem](#)

第 6 次发布

发布时间: 2023-11-21 01:38:03

本次发布包含了以下内容:

改善已有的文档。

新增接口:

- [ImportSourceClusterConsumerGroups](#)
- [ImportSourceClusterTopics](#)

新增数据结构:

- [SourceClusterGroupConfig](#)
- [SourceClusterTopicConfig](#)

第 5 次发布

发布时间: 2023-11-13 00:24:24

本次发布包含了以下内容:

改善已有的文档。

新增接口:

- [DescribeConsumerGroupList](#)

新增数据结构:

- [ConsumeGroupItem](#)

第 4 次发布

发布时间: 2023-09-26 01:33:40

本次发布包含了以下内容:

改善已有的文档。

新增接口:

- [CreateRole](#)
- [DeleteRole](#)
- [DescribeRoleList](#)
- [ModifyRole](#)

新增数据结构:

- [RoleItem](#)

第 3 次发布

发布时间：2023-09-12 02:49:45

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- [CreateConsumerGroup](#)
- [CreateTopic](#)
- [DeleteConsumerGroup](#)
- [DeleteTopic](#)
- [DescribeConsumerGroup](#)
- [DescribeTopic](#)
- [DescribeTopicStatsOp](#)
- [ModifyConsumerGroup](#)
- [ModifyTopic](#)

新增数据结构：

- [SubscriptionData](#)

第 2 次发布

发布时间：2023-09-08 02:48:21

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- [CreateInstance](#)
- [DeleteInstance](#)
- [DescribeInstance](#)
- [ModifyInstance](#)

新增数据结构：

- [Endpoint](#)
- [IpRule](#)
- [VpcInfo](#)

第 1 次发布

发布时间：2023-08-07 10:45:38

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- [DescribeInstanceList](#)
- [DescribeTopicList](#)

新增数据结构：

- [Filter](#)
- [InstanceItem](#)

-
- [Tag](#)
 - [TagFilter](#)
 - [TopicItem](#)

简介

最近更新时间：2023-08-07 10:45:46

欢迎使用 消息队列 RocketMQ 版 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](#)

消息队列 RocketMQ 版（TDMQ for RocketMQ，简称 TDMQ RocketMQ 版）是腾讯云基于 Apache RocketMQ 构建的分布式消息中间件，完全兼容 Apache RocketMQ 的各个组件与概念，支持开源社区版本的客户端零改造接入。

消息队列 RocketMQ 版具有低延迟、高性能、高可靠、万亿级消息容量和灵活可扩展等特点，可为分布式应用系统提供异步解耦和削峰填谷的能力，同时也具备互联网应用所需的海量消息堆积、高吞吐、可靠重试等特性。

API 概览

最近更新时间：2024-05-16 02:34:16

主题相关接口

接口名称	接口功能	频率限制（次/秒）
CreateTopic	创建主题	20
DeleteTopic	删除主题	20
ModifyTopic	修改主题属性	20
DescribeTopicList	查询主题列表	20
DescribeTopic	查询主题详情	20

消费组相关接口

接口名称	接口功能	频率限制（次/秒）
CreateConsumerGroup	创建消费组	10000
DeleteConsumerGroup	删除消费组	10000
ModifyConsumerGroup	修改消费组属性	10000
DescribeConsumerGroup	查询消费组详情	10000
DescribeConsumerGroupList	查询消费组列表	10000

角色鉴权相关接口

接口名称	接口功能	频率限制（次/秒）
CreateRole	添加角色	20
DescribeRoleList	查询角色列表	20
ModifyRole	修改角色	20
DeleteRole	删除角色	20

平滑迁移相关接口

接口名称	接口功能	频率限制（次/秒）
ImportSourceClusterConsumerGroups	平滑迁移：导入消费者组列表	20
ImportSourceClusterTopics	平滑迁移：导入topic列表	20

MQTT相关接口

接口名称	接口功能	频率限制（次/秒）
------	------	-----------

接口名称	接口功能	频率限制（次/秒）
DescribeMQTTInstanceList	获取MQTT实例列表	20
DescribeMQTTInstance	查询MQTT实例详情信息	20
CreateMQTTTopic	创建MQTT主题	20
DeleteMQTTTopic	删除MQTT主题	20
ModifyMQTTTopic	修改MQTT主题属性	20
DescribeMQTTTopicList	查询MQTT主题列表	20
DescribeMQTTMessageList	查询MQTT消息列表	20
CreateMQTTUser	添加MQTT角色	20
DescribeMQTTUserList	查询MQTT用户列表	20
ModifyMQTTUser	修改MQTT角色	20
DeleteMQTTUser	删除MQTT角色	20
DescribeMQTTProductSKUList	获取MQTT产品售卖规格	20
CreateMQTTInstance	购买MQTT实例	20
DeleteMQTTInstance	删除MQTT实例	20
ModifyMQTTInstance	修改MQTT实例属性	20
DescribeMQTTTopic	查询MQTT主题详情	20
ModifyMQTTInstanceCertBinding	更新MQTT集群证书证书	20
DescribeMQTTInstanceCert	查询MQTT集群证书列表	20
DescribeMQTTClient	查询MQTT客户端详情	20
CreateMQTTInsPublicEndpoint	为MQTT实例创建公网接入点	20
DeleteMQTTInsPublicEndpoint	删除MQTT实例的公网接入点	20
DescribeMQTTInsPublicEndpoints	查询MQTT实例公网接入点	20
DescribeMQTTInsVPCEndpoints	查询MQTT实例VPC接入点	20
ModifyMQTTInsPublicEndpoint	更新MQTT实例公网接入点	20
DescribeMQTTMessage	查询MQTT消息	20

集群相关接口

接口名称	接口功能	频率限制（次/秒）
CreateInstance	购买实例	20
DeleteInstance	删除实例	20
ModifyInstance	修改实例属性	20
DescribeInstanceList	获取实例列表	20

接口名称	接口功能	频率限制（次/秒）
DescribeInstance	查询实例信息	20

调用方式

请求结构

最近更新时间：2023-08-07 10:45:46

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 `trocket.tencentcloudapi.com`，也支持指定地域域名访问，例如广州地域的域名为 `trocket.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 `trocket.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）	<code>trocket.tencentcloudapi.com</code>
华南地区(广州)	<code>trocket.ap-guangzhou.tencentcloudapi.com</code>
华东地区(上海)	<code>trocket.ap-shanghai.tencentcloudapi.com</code>
华北地区(北京)	<code>trocket.ap-beijing.tencentcloudapi.com</code>
西南地区(成都)	<code>trocket.ap-chengdu.tencentcloudapi.com</code>
西南地区(重庆)	<code>trocket.ap-chongqing.tencentcloudapi.com</code>
港澳台地区(中国香港)	<code>trocket.ap-hongkong.tencentcloudapi.com</code>
亚太东南(新加坡)	<code>trocket.ap-singapore.tencentcloudapi.com</code>
亚太东南(曼谷)	<code>trocket.ap-bangkok.tencentcloudapi.com</code>
亚太南部(孟买)	<code>trocket.ap-mumbai.tencentcloudapi.com</code>
亚太东北(首尔)	<code>trocket.ap-seoul.tencentcloudapi.com</code>
亚太东北(东京)	<code>trocket.ap-tokyo.tencentcloudapi.com</code>
美国东部(弗吉尼亚)	<code>trocket.na-ashburn.tencentcloudapi.com</code>
美国西部(硅谷)	<code>trocket.na-siliconvalley.tencentcloudapi.com</code>
北美地区(多伦多)	<code>trocket.na-toronto.tencentcloudapi.com</code>
欧洲地区(法兰克福)	<code>trocket.eu-frankfurt.tencentcloudapi.com</code>

注意：由于金融区和非金融区是隔离不互通的，因此当访问金融区服务时（公共参数 Region 为金融区地域），需要同时指定带金融区地域的域名，最好和 Region 的地域保持一致。

金融区接入地域	金融区域名
华东地区(上海金融)	<code>trocket.ap-shanghai-fsi.tencentcloudapi.com</code>

金融区接入地域	金融区域名
华南地区(深圳金融)	trocket.ap-shenzhen-fsi.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：2024-02-23 01:25:56

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：出于简化的目的，部分接口文档中的示例使用的是签名方法 v1 GET 请求，而不是更安全的签名方法 v3。

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。取值参考接口文档输入参数章节关于公共参数 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。取值参考接口文档中输入参数章节关于公共参数 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输入公共参数 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 为具体产品名，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 trocket；tc3_request 为固定字符串； - 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章节关于公共参数 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输入公共参数 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
```

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
亚太东南（曼谷）	ap-bangkok
华北地区（北京）	ap-beijing
西南地区（成都）	ap-chengdu
西南地区（重庆）	ap-chongqing
华南地区（广州）	ap-guangzhou
港澳台地区（中国香港）	ap-hongkong
亚太东南（雅加达）	ap-jakarta
亚太南部（孟买）	ap-mumbai
华东地区（南京）	ap-nanjing
亚太东北（首尔）	ap-seoul
华东地区（上海）	ap-shanghai
华东地区（上海金融）	ap-shanghai-fsi
华南地区（深圳金融）	ap-shenzhen-fsi
亚太东南（新加坡）	ap-singapore
亚太东北（东京）	ap-tokyo
欧洲地区（法兰克福）	eu-frankfurt
美国东部（弗吉尼亚）	na-ashburn
美国西部（硅谷）	na-siliconvalley

签名方法 v3

最近更新時間：2023-12-27 01:36:06

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”，可以对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 8 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++](#)、[Ruby](#)。

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。如已泄露，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】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;x-tc-action, Signature=be4f67d323c78ab9acb7395e43c0dbcf822a9cfac32fea2449a7bc7726b770a3" \
-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 编码（特殊字符编码后需大写字母），字符集 UTF-8。推荐使用编程语言标准库进行编码。

字段名称	解释
CanonicalHeaders	<p>参与签名的头部信息，至少包含 host 和 content-type 两个头部，也可加入其他头部参与签名以提高自身请求的唯一性和安全性，此示例额外增加了接口名头部。</p> <p>拼接规则：</p> <ol style="list-style-type: none"> 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。 <p>此示例计算结果是 content-type:application/json; charset=utf-8\nhost:cvm.tencentcloudapi.com\nx-tc-action:describeinstances\n。</p> <p>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。</p>
SignedHeaders	<p>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。</p> <p>拼接规则：</p> <ol style="list-style-type: none"> 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。 <p>此示例为 content-type;host;x-tc-action</p>
HashedRequestPayload	<p>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。</p>

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
x-tc-action:describeinstances

content-type;host;x-tc-action
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 7019a55be8395899b900fb5564e4200d984910f34794a27cb3fb7d10ff6a1e84。

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，可能运行一段时间后，请求失败，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
7019a55be8395899b900fb5564e4200d984910f34794a27cb3fb7d10ff6a1e84
```

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，可能包含不可打印字符，将其转为十六进制字符串打印的输出分别为：f1cb4d518a0eda9d5cbbfdb7850983f1e603eeae484edea76e4dd8d8deb5556e，e7c609ce81bea53546bed2cc904778bef9ca14082e48e67883443ed64e227cd7，8aa8ab5755582f576e94bcfe383b8e29325b0ca90c3590d569221c6a63a091ed。

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。

字段名称	解释
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 be4f67d323c78ab9acb7395e43c0dbcf822a9cfac32fea2449a7bc7726b770a3。

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;x-tc-action。
Signature	签名值。此示例计算结果是 be4f67d323c78ab9acb7395e43c0dbcf822a9cfac32fea2449a7bc7726b770a3。

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=content-type;host;x-tc-action, Signature=be4f67d323c78ab9acb7395e43c0dbcf822a9cfac32fea2449a7bc7726b770a3
```

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, SignedHeaders=content-type;host;x-tc-action, Signature=be4f67d323c78ab9acb7395e43c0dbcf822a9cfac32fea2449a7bc7726b770a3
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"}]}
```

注意：

请求发送时的 HTTP 头部（Header）和请求体（Payload）必须和签名计算过程中的内容完全一致，否则会返回签名不一致错误。可以通过打印实际请求内容，网络抓包等方式对比排查。

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++](#)
- [Ruby](#)

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](#)

为了更清楚地解释签名过程，下面以实际编程语言为例，将上述的签名过程完整实现。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;
    // 需要设置环境变量 TENCENTCLOUD_SECRET_ID，值为示例的 AKIDz8krbsJ5yKBZQpn74WFkmLPx3*****
    private final static String SECRET_ID = System.getenv("TENCENTCLOUD_SECRET_ID");
    // 需要设置环境变量 TENCENTCLOUD_SECRET_KEY，值为示例的 Gu5t9xGARNpq86cd98joQYCN3*****
    private final static String SECRET_KEY = System.getenv("TENCENTCLOUD_SECRET_KEY");
    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" + "x-tc-action:" + action.toLowerCase() + "\n";
    String signedHeaders = "content-type;host;x-tc-action";

    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
# 需要设置环境变量 TENCENTCLOUD_SECRET_ID, 值为示例的 AKIDz8krbsJ5yKBZQpn74WFkmLPx3*****
secret_id = os.environ.get("TENCENTCLOUD_SECRET_ID")
# 需要设置环境变量 TENCENTCLOUD_SECRET_KEY, 值为示例的 Gu5t9xGARNpq86cd98joQYCN3*****
secret_key = os.environ.get("TENCENTCLOUD_SECRET_KEY")

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.utcnow().timestamp()
params = {"Limit": 1, "Filters": [{"Values": [u"未命名"], "Name": "instance-name"}]}

# ***** 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\nx-tc-action:%s\n" % (ct, host, action.lower())
signed_headers = "content-type;host;x-tc-action"
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"  
    "os"  
    "strings"  
    "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() {  
    // 需要设置环境变量 TENCENTCLOUD_SECRET_ID, 值为示例的 AKIDz8krbsJ5yKBZQpn74WFkmLPx3*****  
    secretId := os.Getenv("TENCENTCLOUD_SECRET_ID")  
    // 需要设置环境变量 TENCENTCLOUD_SECRET_KEY, 值为示例的 Gu5t9xGARNpq86cd98joQYCN3*****  
    secretKey := os.Getenv("TENCENTCLOUD_SECRET_KEY")  
    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 := fmt.Sprintf("content-type:%s\nhost:%s\nx-tc-action:%s\n",
    "application/json; charset=utf-8", host, strings.ToLower(action))
signedHeaders := "content-type;host;x-tc-action"
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
// 需要设置环境变量 TENCENTCLOUD_SECRET_ID, 值为示例的 AKIDz8krbsJ5yKBZQpn74WFkmLPx3*****
$secretId = getenv("TENCENTCLOUD_SECRET_ID");
// 需要设置环境变量 TENCENTCLOUD_SECRET_KEY, 值为示例的 Gu5t9xGARNpq86cd98joQYCN3*****
$secretKey = getenv("TENCENTCLOUD_SECRET_KEY");
$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 = "";
$canonicalHeaders = implode("\n", [
"content-type:application/json; charset=utf-8",
"host:".$host,
"x-tc-action:".strtolower($action),
""
]);
$signedHeaders = implode(";", [
"content-type",
"host",
"x-tc-action",
]);
$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."/". $credentialScope
.", SignedHeaders=".$signedHeaders.", 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
# 需要设置环境变量 TENCENTCLOUD_SECRET_ID, 值为示例的 AKIDz8krbsj5yKBZQpn74WFkmLPx3*****
secret_id = ENV["TENCENTCLOUD_SECRET_ID"]
# 需要设置环境变量 TENCENTCLOUD_SECRET_KEY, 值为示例的 Gu5t9xGARNpq86cd98joQYCN3*****
secret_key = ENV["TENCENTCLOUD_SECRET_KEY"]

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-x-tc-action:#{action.downcase}\n"
signed_headers = 'content-type;host;x-tc-action'
# 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": ["\u6672a\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, 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"
+ "x-tc-action:" + action.ToLower() + "\n";
string signedHeaders = "content-type;host;x-tc-action";
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"
+ hashedCanonicalRequest;
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 = " "
+ "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
// 需要设置环境变量 TENCENTCLOUD_SECRET_ID, 值为示例的 AKIDz8krbsJ5yKBZQpn74WFkmLPx3*****
string SECRET_ID = Environment.GetEnvironmentVariable("TENCENTCLOUD_SECRET_ID");
// 需要设置环境变量 TENCENTCLOUD_SECRET_KEY, 值为示例的 Gu5t9xGARNpq86cd98joQYCN3*****
string SECRET_KEY = Environment.GetEnvironmentVariable("TENCENTCLOUD_SECRET_KEY");

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, 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
    // 需要设置环境变量 TENCENTCLOUD_SECRET_ID, 值为示例的 AKIDz8krbsJ5yKBZQpn74WFkmLPx3*****
    const SECRET_ID = process.env.TENCENTCLOUD_SECRET_ID
    // 需要设置环境变量 TENCENTCLOUD_SECRET_KEY, 值为示例的 Gu5t9xGARNpq86cd98joQYCN3*****
    const SECRET_KEY = process.env.TENCENTCLOUD_SECRET_KEY

    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 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"
    + "x-tc-action:" + action.toLowerCase() + "\n"
    const signedHeaders = "content-type;host;x-tc-action"

    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 <algorithm>
#include <cstdlib>
#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++)
    {
        sprintf(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);

#ifdef OPENSSSL_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
    // 需要设置环境变量 TENCENTCLOUD_SECRET_ID, 值为示例的 AKIDz8krbsJ5yKBZQpn74WFkmLPx3*****
    string SECRET_ID = getenv("TENCENTCLOUD_SECRET_ID");
    // 需要设置环境变量 TENCENTCLOUD_SECRET_KEY, 值为示例的 Gu5t9xGARNpq86cd98joQYCN3*****
    string SECRET_KEY = getenv("TENCENTCLOUD_SECRET_KEY");

    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 lower = action;
std::transform(action.begin(), action.end(), lower.begin(), ::tolower);
string canonicalHeaders = string("content-type:application/json; charset=utf-8\n")
+ "host:" + host + "\n"
+ "x-tc-action:" + lower + "\n";
string signedHeaders = "content-type;host;x-tc-action";
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;
};
```

C

```
#include <ctype.h>
#include <string.h>
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#include <stdint.h>
#include <openssl/sha.h>
#include <openssl/hmac.h>

void get_utc_date(int64_t timestamp, char* utc, int len)
{
    // time_t timenow;
    struct tm sttime;
    sttime = *gmtime(&timestamp);
    strftime(utc, len, "%Y-%m-%d", &sttime);
}

void sha256_hex(const char* str, char* result)
{
    char buf[3];
    unsigned char hash[SHA256_DIGEST_LENGTH];
    SHA256_CTX sha256;
    SHA256_Init(&sha256);
    SHA256_Update(&sha256, str, strlen(str));
    SHA256_Final(hash, &sha256);
    for(int i = 0; i < SHA256_DIGEST_LENGTH; i++)
    {
        sprintf(buf, sizeof(buf), "%02x", hash[i]);
        strcat(result, buf);
    }
}

void hmac_sha256(char* key, const char* input, char* result)
{
    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, strlen(key), EVP_sha256(), NULL);
HMAC_Update(h, ( unsigned char* )input, strlen(input));
unsigned int len = 32;
HMAC_Final(h, hash, &len);

#ifdef OPENSSL_VERSION_NUMBER < 0x10100000L
HMAC_CTX_cleanup(h);
#else
HMAC_CTX_free(h);
#endif
strncpy(result, (const char*)hash, len);
}

void hex_encode(const char* input, char* output)
{
    static const char* const lut = "0123456789abcdef";
    size_t len = strlen(input);
    char add_out[128] = {0};
    char temp[2] = {0};
    for (size_t i = 0; i < len; ++i)
    {
        const unsigned char c = input[i];
        temp[0] = lut[c >> 4];
        strcat(add_out, temp);
        temp[0] = lut[c & 15];
        strcat(add_out, temp);
    }
    strncpy(output, add_out, 128);
}

void lowercase(const char * src, char * dst)
{
    for (int i = 0; src[i]; i++)
    {
        dst[i] = tolower(src[i]);
    }
}

int main()
{
    // 密钥参数
    // 需要设置环境变量 TENCENTCLOUD_SECRET_ID, 值为示例的 AKIDz8krbsJ5yKBZQpn74WFkmLPx3*****
    const char* SECRET_ID = getenv("TENCENTCLOUD_SECRET_ID");
    // 需要设置环境变量 TENCENTCLOUD_SECRET_KEY, 值为示例的 Gu5t9xGARNpq86cd98joQYCN3*****
    const char* SECRET_KEY = getenv("TENCENTCLOUD_SECRET_KEY");
}

```

```

const char* service = "cvm";
const char* host = "cvm.tencentcloudapi.com";
const char* region = "ap-guangzhou";
const char* action = "DescribeInstances";
const char* version = "2017-03-12";
int64_t timestamp = 1551113065;
char date[20] = {0};
get_utc_date(timestamp, date, sizeof(date));

// ***** 步骤 1: 拼接规范请求串 *****
const char* http_request_method = "POST";
const char* canonical_uri = "/";
const char* canonical_query_string = "";
char canonical_headers[100] = {"content-type:application/json; charset=utf-8\nhost:"};
strcat(canonical_headers, host);
strcat(canonical_headers, "\nx-tc-action:");
char value[100] = {0};
lowercase(action, value);
strcat(canonical_headers, value);
strcat(canonical_headers, "\n");
const char* signed_headers = "content-type;host;x-tc-action";
const char* payload = "{\"Limit\": 1, \"Filters\": [{\"Values\": [\"\\u672a\\u547d\\u540d\"], \"Name\": \"instance-name\"}] }";
char hashed_request_payload[100] = {0};
sha256_hex(payload, hashed_request_payload);

char canonical_request[256] = {0};
sprintf(canonical_request, "%s\n%s\n%s\n%s\n%s\n%s", http_request_method,
canonical_uri, canonical_query_string, canonical_headers,
signed_headers, hashed_request_payload);
printf("%s\n", canonical_request);

// ***** 步骤 2: 拼接待签名字符串 *****
const char* algorithm = "TC3-HMAC-SHA256";
char request_timestamp[16] = {0};
sprintf(request_timestamp, "%d", timestamp);
char credential_scope[64] = {0};
strcat(credential_scope, date);
sprintf(credential_scope, "%s/%s/tc3_request", date, service);
char hashed_canonical_request[100] = {0};
sha256_hex(canonical_request, hashed_canonical_request);
char string_to_sign[256] = {0};
sprintf(string_to_sign, "%s\n%s\n%s\n%s", algorithm, request_timestamp,
credential_scope, hashed_canonical_request);
printf("%s\n", string_to_sign);

// ***** 步骤 3: 计算签名 *****
char k_key[64] = {0};
sprintf(k_key, "%s%s", "TC3", SECRET_KEY);
char k_date[64] = {0};
    
```

```

hmac_sha256(k_key, date, k_date);
char k_service[64] = {0};
hmac_sha256(k_date, service, k_service);
char k_signing[64] = {0};
hmac_sha256(k_service, "tc3_request", k_signing);
char k_hmac_sha_sign[64] = {0};
hmac_sha256(k_signing, string_to_sign, k_hmac_sha_sign);

char signature[128] = {0};
hex_encode(k_hmac_sha_sign, signature);
printf("%s\n", signature);

// ***** 步骤 4: 拼接 Authorization *****
char authorization[512] = {0};
sprintf(authorization, "%s Credential=%s/%s, SignedHeaders=%s, Signature=%s",
algorithm, SECRET_ID, credential_scope, signed_headers, signature);
printf("%s\n", authorization);

char curlcmd[10240] = {0};
sprintf(curlcmd, "curl -X POST https://%s\n \
-H \"Authorization: %s\"\n \
-H \"Content-Type: application/json; charset=utf-8\"\n \
-H \"Host: %s\"\n \
-H \"X-TC-Action: %s\"\n \
-H \"X-TC-Timestamp: %s\"\n \
-H \"X-TC-Version: %s\"\n \
-H \"X-TC-Region: %s\"\n \
-d \"%s\"",
host, authorization, host, action, request_timestamp, version, region, payload);
printf("%s\n", curlcmd);
return 0;
}
    
```

其他语言

- Lua: [GitHub](#)
- Swift: [GitHub](#)
- Dart: [GitHub](#)
- Shell(Bash): [GitHub](#)

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。

错误码	错误描述
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：2024-01-05 01:28:16

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 8 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++](#)、[Ruby](#)。

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
InstanceId.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',
  'InstanceId.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“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"，因此格式化后即为 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&InstanceId.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&InstanceId=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&InstanceId=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 编码前需要先用 UTF-8 进行编码。

注意: 有些编程语言的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++](#)
- [Ruby](#)

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&InstanceId.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，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.DataConverter;

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
    // 需要设置环境变量 TENCENTCLOUD_SECRET_ID, 值为示例的 AKIDz8krbsJ5yKBZQpn74WfkmLPx3*****
    params.put("SecretId", System.getenv("TENCENTCLOUD_SECRET_ID")); // 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("InstanceId", "ins-09dx96dg"); // 业务参数
    // 需要设置环境变量 TENCENTCLOUD_SECRET_KEY, 值为示例的 Gu5t9xGARNpq86cd98joQYCN3*****
    params.put("Signature", sign(getStringToSign(params), System.getenv("TENCENTCLOUD_SECRET_KEY"), "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 。

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

import requests

# 需要设置环境变量 TENCENTCLOUD_SECRET_ID，值为示例的 AKIDz8krbsj5yKBZQpn74WFkmLPx3*****
secret_id = os.environ.get("TENCENTCLOUD_SECRET_ID")
# 需要设置环境变量 TENCENTCLOUD_SECRET_KEY，值为示例的 Gu5t9xGARNpq86cd98joQYCN3*****
secret_key = os.environ.get("TENCENTCLOUD_SECRET_KEY")

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',
        'InstanceId.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

```
package main
```

```
import (  
    "bytes"  
    "crypto/hmac"  
    "crypto/sha1"  
    "encoding/base64"  
    "fmt"  
    "os"  
    "sort"  
    "strconv"  
)  
  
func main() {  
    // 需要设置环境变量 TENCENTCLOUD_SECRET_ID, 值为示例的 AKIDz8krbsJ5yKBZQpn74WFkmLPx3*****  
    secretId := os.Getenv("TENCENTCLOUD_SECRET_ID")  
    // 需要设置环境变量 TENCENTCLOUD_SECRET_KEY, 值为示例的 Gu5t9xGARNpq86cd98joQYCN3*****  
    secretKey := os.Getenv("TENCENTCLOUD_SECRET_KEY")  
    params := map[string]string{  
        "Nonce": "11886",  
        "Timestamp": strconv.Itoa(1465185768),  
        "Region": "ap-guangzhou",  
        "SecretId": secretId,  
        "Version": "2017-03-12",  
        "Action": "DescribeInstances",  
        "InstanceId.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
// 需要设置环境变量 TENCENTCLOUD_SECRET_ID, 值为示例的 AKIDz8krbsJ5yKBZQpn74WFkmLPx3*****
$secretId = getenv("TENCENTCLOUD_SECRET_ID");
// 需要设置环境变量 TENCENTCLOUD_SECRET_KEY, 值为示例的 Gu5t9xGARNpq86cd98joQYCN3*****
$secretKey = getenv("TENCENTCLOUD_SECRET_KEY");
$params["Nonce"] = 11886;//rand();
$params["Timestamp"] = 1465185768;//time();
$params["Region"] = "ap-guangzhou";
$params["SecretId"] = $secretId;
$params["Version"] = "2017-03-12";
$params["Action"] = "DescribeInstances";
$params["InstanceId.0"] = "ins-09dx96dg";
$params["Limit"] = 20;
$params["Offset"] = 0;

ksort($params);

$signStr = "GETcvm.tencentcloudapi.com/?";
foreach ( $params 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
// $params["Signature"] = $signature;
// $url = "https://cvm.tencentcloudapi.com/?".http_build_query($params);
// 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'

# 需要设置环境变量 TENCENTCLOUD_SECRET_ID, 值为示例的 AKIDz8krbsJ5yKBZQpn74WFkmLPx3*****
secret_id = ENV["TENCENTCLOUD_SECRET_ID"]
# 需要设置环境变量 TENCENTCLOUD_SECRET_KEY, 值为示例的 Gu5t9xGARNpq86cd98joQYCN3*****
secret_key = ENV["TENCENTCLOUD_SECRET_KEY"]

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 requestHost, 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
    // 需要设置环境变量 TENCENTCLOUD_SECRET_ID, 值为示例的 AKIDz8krbsJ5yKBZQpn74WFkmLPx3*****
    string SECRET_ID = Environment.GetEnvironmentVariable("TENCENTCLOUD_SECRET_ID");
    // 需要设置环境变量 TENCENTCLOUD_SECRET_KEY, 值为示例的 Gu5t9xGARNpq86cd98joQYCN3*****
    string SECRET_KEY = Environment.GetEnvironmentVariable("TENCENTCLOUD_SECRET_KEY");

    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("InstanceId.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'] = encodeURIComponent(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
    // 需要设置环境变量 TENCENTCLOUD_SECRET_ID, 值为示例的 AKIDz8krbsJ5yKBZQpn74WFkmLPx3*****
    const SECRET_ID = process.env.TENCENTCLOUD_SECRET_ID
    // 需要设置环境变量 TENCENTCLOUD_SECRET_KEY, 值为示例的 Gu5t9xGARNpq86cd98joQYCN3*****
    const SECRET_KEY = process.env.TENCENTCLOUD_SECRET_KEY

    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['InstanceId.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：2024-03-12 01:58:25

云 API 3.0 接口默认返回 JSON 数据，返回非 JSON 格式的接口会在文档中做出说明。返回 JSON 数据时最大限制为 50 MB，如果返回的数据超过最大限制，请求会失败并返回内部错误。请根据接口文档中给出的过滤功能（例如时间范围）或者分页功能，控制返回数据不要过大。

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 [腾讯云客服](#) 或 [提交工单](#)，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 [腾讯云客服](#) 或 [提交工单](#)，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：2023-08-07 10:45:48

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的整型定义，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。

主题相关接口

创建主题

最近更新时间：2024-03-12 01:58:22

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

创建主题

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：CreateTopic。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
Instanceid	是	String	实例ID 示例值：rocketmq-xxxx
Topic	是	String	主题 示例值：test
TopicType	是	String	主题类型 UNSPECIFIED:未指定， NORMAL:普通消息， FIFO:顺序消息， DELAY:延时消息， TRANSACTION:事务消息
QueueNum	是	Integer	队列数量 示例值：3
Remark	否	String	备注
MsgTTL	否	Integer	消息保留时长 示例值：72

3. 输出参数

参数名称	类型	描述
Instanceid	String	实例ID

参数名称	类型	描述
Topic	String	主题
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 创建主题

创建主题

输入示例

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

{
  "InstanceId": "rmq-72mo3a9o",
  "Topic": "test",
  "TopicType": "NORMAL",
  "QueueNum": 16,
  "Remark": "remoark"
}
```

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "InstanceId": "rmq-72mo3a9o",
    "RequestId": "97f45511-b653-4e2a-ade6-58c8d3ae523b",
    "Topic": "test"
  }
}
```

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: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub Gitee](#)

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	操作失败。
FailedOperation.InstanceNotReady	实例未就绪，请稍后重试。
LimitExceeded.TopicNum	已达到主题数上限。
OperationDenied	操作被拒绝。
ResourceNotFound.Instance	实例不存在。
UnsupportedOperation.ResourceAlreadyExists	资源已存在，请检查后重试。

删除主题

最近更新时间：2024-03-12 01:58:22

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

删除主题

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：DeleteTopic。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
InstanceId	是	String	实例ID 示例值：rocketmq-xxxx
Topic	是	String	主题 示例值：test

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 删除主题

删除主题

输入示例

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

```
{
  "InstanceId": "rmq-72mo3a9o",
  "Topic": "test"
}
```

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "RequestId": "4687272f-f35c-4aef-9941-ced81c164d4b"
  }
}
```

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: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub Gitee](#)

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	操作失败。
ResourceNotFound.Topic	主题不存在，请检查后重试。

修改主题属性

最近更新时间：2024-04-04 01:25:27

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

修改主题属性

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：ModifyTopic。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
InstanceId	是	String	实例ID 示例值：rocketmq-xxxx
Topic	是	String	主题 示例值：test
QueueNum	否	Integer	队列数量 示例值：6
Remark	否	String	备注信息 示例值：test
MsgTTL	否	Integer	消息保留时长 示例值：72

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 修改主题属性

修改主题属性

输入示例

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

{
  "InstanceId": "rmq-72mo3a9o",
  "Topic": "test",
  "QueueNum": 1
}
```

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "RequestId": "945afc69-1c24-4891-a2ce-c5fa6032ba75"
  }
}
```

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: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub Gitee](#)

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	操作失败。

查询主题列表

最近更新时间：2024-03-12 01:58:21

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

获取主题列表，Filter参数使用说明如下：

1. TopicName，主题名称模糊搜索
2. TopicType，主题类型查询，支持多选，可选值：Normal,Order,Transaction,DelayScheduled

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：DescribeTopicList。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
InstanceId	是	String	实例ID 示例值：rocketmq-xxxx
Filters.N	否	Array of Filter	查询条件列表
Offset	否	Integer	查询起始位置 示例值：0
Limit	否	Integer	查询结果限制数量 示例值：10

3. 输出参数

参数名称	类型	描述
TotalCount	Integer	查询总数 注意：此字段可能返回 null，表示取不到有效值。 示例值：10
Data	Array of TopicItem	主题列表
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 查询主题列表

查询主题列表

输入示例

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

{
  "InstanceId": "rmq-72mo3a9o",
  "Offset": 0,
  "Limit": 1
}
```

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "Data": [
      {
        "InstanceId": "rmq-72mo3a9o",
        "QueueNum": 16,
        "Remark": "remark",
        "Topic": "test",
        "TopicType": "NORMAL"
      }
    ],
    "RequestId": "927cbd2c-be83-410e-a83f-24dcea1d88d9",
    "TotalCount": 7
  }
}
```

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: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub](#) [Gitee](#)

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
ResourceNotFound.Instance	实例不存在。

查询主题详情

最近更新时间：2024-03-12 01:58:21

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

查询主题详情，Offset和Limit参数是指订阅该主题的消费组查询分页参数，Filter参数使用说明如下：

ConsumerGroup，消费组名称过滤

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：DescribeTopic。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
InstanceId	是	String	实例ID 示例值：rocketmq-xxxx
Topic	是	String	主题 示例值：test
Filters.N	否	Array of Filter	查询条件列表
Offset	否	Integer	查询起始位置 示例值：0
Limit	否	Integer	查询结果限制数量 示例值：10

3. 输出参数

参数名称	类型	描述
InstanceId	String	实例ID
Topic	String	主题名称

参数名称	类型	描述
TopicType	String	主题类型 UNSPECIFIED:未指定, NORMAL:普通消息, FIFO:顺序消息, DELAY:延时消息, TRANSACTION:事务消息
Remark	String	备注
CreateTime	Integer	创建时间, 秒为单位
LastUpdateTime	Integer	最后写入时间, 秒为单位
SubscriptionCount	Integer	订阅数量
SubscriptionData	Array of SubscriptionData	订阅关系列表
MsgTTL	Integer	消息保留时长 示例值: 72
RequestId	String	唯一请求 ID, 由服务端生成, 每次请求都会返回 (若请求因其他原因未能抵达服务端, 则该次请求不会获得 RequestId)。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 查询主题详情

查询主题详情

输入示例

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

{
  "InstanceId": "rmq-72mo3a9o",
  "Topic": "test",
  "Offset": 0,
  "Limit": 10
}
```

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "CreateTime": 1683601674000,
    "InstanceId": "rmq-72mo3a9o",
```

```
"LastUpdateTime": 0,
"Remark": "remoark",
"RequestId": "f7f7ba8a-4335-4adb-9186-680b91502c42",
"SubscriptionCount": 0,
"SubscriptionData": [],
"Topic": "test",
"TopicType": "NORMAL"
}
}
```

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: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub Gitee](#)

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
ResourceNotFound.Instance	实例不存在。
ResourceNotFound.Topic	主题不存在，请检查后重试。

消费组相关接口

创建消费组

最近更新时间：2024-03-12 01:58:20

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

创建消费组

默认接口请求频率限制：10000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：CreateConsumerGroup。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
InstanceId	是	String	实例ID 示例值：rocketmq-xxxx
ConsumerGroup	是	String	消费组名称 示例值：group_test
MaxRetryTimes	是	Integer	最大重试次数 示例值：16
ConsumeEnable	是	Boolean	是否开启消费
ConsumeMessageOrderly	是	Boolean	顺序投递：true 并发投递：false
Remark	否	String	备注

3. 输出参数

参数名称	类型	描述
InstanceId	String	实例ID
ConsumerGroup	String	消费组
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 创建消费组

创建消费组

输入示例

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

{
  "InstanceId": "rmq-72mo3a9o",
  "ConsumerGroup": "group",
  "MaxRetryTimes": 16,
  "ConsumeEnable": true,
  "ConsumeMessageOrderly": true,
  "Remark": "test"
}
```

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "ConsumerGroup": "group",
    "InstanceId": "rmq-72mo3a9o",
    "RequestId": "58f60891-002f-4b8a-b71f-3767a645c28e"
  }
}
```

示例2 创建消费者组

创建消费者组

输入示例

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

{
  "InstanceId": "rmq-g83xq2jn",
  "ConsumerGroup": "group",
  "ConsumeMessageOrderly": false,
}
```

```
"MaxRetryTimes": 16,
"ConsumeEnable": true
}
```

输出示例

```
{
  "Response": {
    "RequestId": "60a1347e-e418-42f1-9f9e-00226a3a3c18",
    "InstanceId": "rmq-g83xq2jn",
    "ConsumerGroup": "group"
  }
}
```

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: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub Gitee](#)

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	操作失败。
FailedOperation.InstanceNotReady	实例未就绪，请稍后重试。
ResourceNotFound.Instance	实例不存在。

错误码	描述
UnsupportedOperation.ResourceAlreadyExists	资源已存在，请检查后重试。

删除消费组

最近更新时间：2024-03-12 01:58:20

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

删除消费组

默认接口请求频率限制：10000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：DeleteConsumerGroup。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
InstanceId	是	String	实例ID 示例值：rocketmq-xxxx
ConsumerGroup	是	String	消费组名称 示例值：group_test

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 删除消费组

删除消费组

输入示例

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

```
{
  "InstanceId": "rmq-72mo3a9o",
  "ConsumerGroup": "group"
}
```

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "RequestId": "8221cd04-dafc-438d-bc1f-816fd87df24e"
  }
}
```

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: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub Gitee](#)

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	操作失败。
ResourceNotFound.Group	消费组不存在，请检查后重试。

修改消费组属性

最近更新时间：2024-03-12 01:58:19

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

修改消费组属性

默认接口请求频率限制：10000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：ModifyConsumerGroup。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
InstanceId	是	String	实例ID 示例值：rocketmq-xxxx
ConsumerGroup	否	String	消费组名称 示例值：group_test
ConsumeEnable	否	Boolean	是否开启消费
ConsumeMessageOrderly	否	Boolean	顺序投递：true 并发投递：false
MaxRetryTimes	否	Integer	最大重试次数
Remark	否	String	备注

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 修改消费组属性

修改消费组属性

输入示例

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

```
{
  "InstanceId": "rmq-72mo3a9o",
  "ConsumerGroup": "group",
  "ConsumeEnable": true,
  "ConsumeMessageOrderly": true,
  "MaxRetryTimes": 16,
  "Remark": "test"
}
```

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "RequestId": "239a3769-8dc6-41e9-bc01-67be452db5a3"
  }
}
```

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: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub](#) [Gitee](#)

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

该接口暂无业务逻辑相关的错误码，其他错误码详见 [公共错误码](#)。

查询消费组详情

最近更新时间：2024-03-12 01:58:19

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

查询消费组详情

默认接口请求频率限制：10000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：DescribeConsumerGroup。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
InstanceId	是	String	实例ID 示例值：rocketmq-xxxx
ConsumerGroup	否	String	消费组名称 示例值：group_test

3. 输出参数

参数名称	类型	描述
ConsumerNum	Integer	在线消费者数量
Tps	Integer	TPS
ConsumerLag	Integer	消息堆积数量
ConsumeType	String	消费者类型
CreatedTime	Integer	创建时间，秒为单位
ConsumeMessageOrderly	Boolean	顺序投递：true 并发投递：false
ConsumeEnable	Boolean	是否开启消费
MaxRetryTimes	Integer	最大重试次数
Remark	String	备注

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 查询消费组详情

查询消费者详情

输入示例

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

{
  "InstanceId": "rmq-72mo3a9o",
  "ConsumerGroup": "group"
}
```

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "ConsumeEnable": true,
    "ConsumeMessageOrderly": true,
    "ConsumeType": "",
    "ConsumerLag": 0,
    "ConsumerNum": 0,
    "CreatedTime": 1683613488000,
    "MaxRetryTimes": 16,
    "Remark": "test",
    "RequestId": "bc4ffab1-65cc-4ef6-9114-0de06c333b84",
    "Tps": 0
  }
}
```

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: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub Gitee](#)

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
ResourceNotFound.Group	消费组不存在，请检查后重试。
ResourceNotFound.Instance	实例不存在。

查询消费组列表

最近更新时间：2024-03-12 01:58:19

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

获取消费组列表，Filter参数使用说明如下：

1. ConsumerGroupName，名称模糊查询
2. ConsumeMessageOrderly，投递顺序性。"true":顺序投递；"false":并发投递

默认接口请求频率限制：10000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：DescribeConsumerGroupList。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
InstanceId	是	String	实例ID 示例值：rocketmq-xxxx
Offset	是	Integer	查询起始位置 示例值：0
Limit	是	Integer	查询结果限制数量 示例值：10
Filters.N	否	Array of Filter	查询条件列表
FromTopic	否	String	查询指定主题下的消费组

3. 输出参数

参数名称	类型	描述
TotalCount	Integer	查询总数 注意：此字段可能返回 null，表示取不到有效值。 示例值：10
Data	Array of ConsumeGroupItem	消费组列表

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 获取消费组列表

获取消费组列表

输入示例

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

{
  "InstanceId": "rmq-72mo3a9o",
  "Offset": 0,
  "Limit": 1
}
```

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "Data": [
      {
        "ConsumeEnable": true,
        "ConsumeMessageOrderly": true,
        "ConsumerGroup": "group",
        "InstanceId": "rmq-72mo3a9o",
        "MaxRetryTimes": 16,
        "Remark": "test"
      }
    ],
    "RequestId": "318142bf-0dd1-4b81-9444-a40f988619a1",
    "TotalCount": 7
  }
}
```

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: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub Gitee](#)

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
ResourceNotFound.Instance	实例不存在。

角色鉴权相关接口

添加角色

最近更新时间：2024-03-12 01:58:19

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

添加角色

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：CreateRole。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
Instanceld	是	String	实例ID 示例值：rocketmq-xxxx
Role	是	String	角色名称
Remark	是	String	备注
PermWrite	是	Boolean	是否开启生产权限
PermRead	是	Boolean	是否开启消费权限

3. 输出参数

参数名称	类型	描述
Role	String	角色名
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 添加角色

添加角色

输入示例

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

{
  "InstanceId": "rmq-72mo3a9o",
  "Role": "testRole",
  "Remark": "testRemark",
  "PermWrite": true,
  "PermRead": true
}
```

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "RequestId": "0bb79e96-2531-4088-8d90-e1372c15518a",
    "Role": "testRole"
  }
}
```

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: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub](#) [Gitee](#)

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	操作失败。
FailedOperation.InstanceNotReady	实例未就绪，请稍后重试。
OperationDenied	操作被拒绝。
ResourceInUse	资源被占用。
ResourceNotFound.Instance	实例不存在。
UnsupportedOperation.ResourceAlreadyExists	资源已存在，请检查后重试。

查询角色列表

最近更新时间：2024-03-12 01:58:18

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

查询角色列表，Filter参数使用说明如下：

1. RoleName，角色名称模糊搜索

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：DescribeRoleList。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
Instanceld	是	String	实例ID 示例值：rocketmq-xxxx
Offset	是	Integer	查询起始位置 示例值：0
Limit	是	Integer	查询结果限制数量 示例值：10
Filters.N	否	Array of Filter	查询条件列表

3. 输出参数

参数名称	类型	描述
TotalCount	Integer	查询总数 注意：此字段可能返回 null，表示取不到有效值。 示例值：10
Data	Array of RoleItem	角色信息列表
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 查询角色列表

查询角色列表

输入示例

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

{
  "InstanceId": "rmq-72mo3a9o",
  "Offset": 0,
  "Limit": 1
}
```

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "Data": [
      {
        "AccessKey": "eyJrZXIjZCI6InJtcS03Mm1vM2E5byIsImFsZyI6IkhTMjU2In0.eyJzdWl0IjpbXEtNzJtbzNhOW9fdGVzdFJvbGUifQ.vrdHDz
zfeDsX1c5rhaFcLpJBd_dGIWQIIAPLOuOEDfM",
        "CreatedTime": 1683615280000,
        "ModifiedTime": 1683615280000,
        "PermRead": true,
        "PermWrite": true,
        "Remark": "testRemark",
        "RoleName": "testRole",
        "SecretKey": "/tuS7zTKFNmsApaUvq4I5kTnV/GSWJ2m4bGhR62kVYY="
      }
    ],
    "RequestId": "346f63a2-dd66-439e-9abc-338a56f6523c",
    "TotalCount": 2
  }
}
```

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: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub](#) [Gitee](#)

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

该接口暂无业务逻辑相关的错误码，其他错误码详见 [公共错误码](#)。

修改角色

最近更新时间：2024-03-12 01:58:18

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

修改角色

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：ModifyRole。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
InstanceId	是	String	实例ID 示例值：rocketmq-xxxx
Role	是	String	角色名称
PermRead	是	Boolean	是否开启消费
PermWrite	是	Boolean	是否开启生产
Remark	否	String	备注

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 修改角色

修改角色

输入示例

```
POST / HTTP/1.1
```

```
Host: trocket.tencentcloudapi.com
```

```
Content-Type: application/json
```

```
X-TC-Action: ModifyRole
```

```
<公共请求参数>
```

```
{
  "InstanceId": "rmq-72mo3a9o",
  "Role": "testRole",
  "Remark": "test",
  "PermRead": true,
  "PermWrite": true
}
```

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "RequestId": "0608515b-cde2-41d6-933c-169d0d1849c5"
  }
}
```

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: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub](#) [Gitee](#)

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	操作失败。

删除角色

最近更新时间：2024-03-12 01:58:19

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

删除角色

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：DeleteRole。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
InstanceId	是	String	实例ID 示例值：rocketmq-xxxx
Role	是	String	角色名称

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 删除角色

删除角色

输入示例

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

```
{
  "InstanceId": "rmq-72mo3a9o",
  "Role": "testRole"
}
```

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "RequestId": "c6e5472a-231d-4bd0-84cf-447f76a8216e"
  }
}
```

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: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub Gitee](#)

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	操作失败。
ResourceNotFound.Instance	实例不存在。
ResourceNotFound.Role	角色不存在，请检查后重试。

平滑迁移相关接口

平滑迁移：导入消费者组列表

最近更新时间：2024-03-12 01:58:20

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

导入消费者组列表

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： ImportSourceClusterConsumerGroups。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
TaskId	是	String	任务ID
GroupList.N	是	Array of SourceClusterGroupConfig	待导入的消费组列表

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 导入源集群消费组

输入示例

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

```
{
  "TaskId": "abc",
  "GroupList": [
    {
      "GroupName": "Test",
      "Remark": "abc",
      "Namespace": ""
    }
  ]
}
```

输出示例

```
{
  "Response": {
    "RequestId": "abc"
  }
}
```

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: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub Gitee](#)

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

该接口暂无业务逻辑相关的错误码，其他错误码详见 [公共错误码](#)。

平滑迁移：导入topic列表

最近更新时间：2024-03-12 01:58:20

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

导入topic列表

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：ImportSourceClusterTopics。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
TaskId	是	String	任务ID
TopicList.N	是	Array of SourceClusterTopicConfig	待导入的主题列表

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 导入源集群主题列表

输入示例

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

{
  "TaskId": "abc",
```

```
"TopicList": [  
  {  
    "TopicName": "Test",  
    "TopicType": "Normal",  
    "QueueNum": 10,  
    "Remark": "abc",  
    "Namespace": ""  
  }  
]
```

输出示例

```
{  
  "Response": {  
    "RequestId": "abc"  
  }  
}
```

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: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub Gitee](#)

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

该接口暂无业务逻辑相关的错误码，其他错误码详见 [公共错误码](#)。

MQTT相关接口

获取MQTT实例列表

最近更新时间：2024-04-22 01:28:22

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

获取实例列表，Filters参数使用说明如下：

1. InstanceName, 名称模糊查询
2. InstanceId, 实例ID查询
3. InstanceType, 实例类型查询，支持多选
4. InstanceStatus, 实例状态查询，支持多选

当使用TagFilters查询时，Filters参数失效。

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：DescribeMQTTInstanceList。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
Filters.N	否	Array of Filter	查询条件列表
Offset	否	Integer	查询起始位置 示例值：0
Limit	否	Integer	查询结果限制数量 示例值：10

3. 输出参数

参数名称	类型	描述
TotalCount	Integer	查询总数 注意：此字段可能返回 null，表示取不到有效值。 示例值：10

参数名称	类型	描述
Data	Array of MQTTInstanceItem	实例列表 示例值: [{ "ClientNumLimit": 1, "CreateTime": 1705565760, "InstanceId": "mqtt-wjgxr8jg", "InstanceName": "su-0118-no-pub", "InstanceStatus": "CREATE_FAILURE", "InstanceType": "BASIC", "Remark": "", "SkuCode": "mqtt_1k", "SubscriptionNumLimit": 1, "TopicNum": 0, "TopicNumLimit": 1, "TpsLimit": 1, "Version": "1" }, { "ClientNumLimit": 1000, "CreateTime": 1705391046, "InstanceId": "mqtt-47ka4rdr", "InstanceName": "sun-0116", "InstanceStatus": "RUNNING", "InstanceType": "BASIC", "Remark": "", "SkuCode": "basic_1k", "SubscriptionNumLimit": 1000, "TopicNum": 25, "TopicNumLimit": 25, "TpsLimit": 1000, "Version": "1" }]
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 示例

示例

输入示例

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

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

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "Data": [
      {
        "ClientNumLimit": 1,
        "CreateTime": 1705565760,
        "InstanceId": "mqtt-wjgxr8jg",
        "InstanceName": "su-0118-no-pub",
        "InstanceStatus": "CREATE_FAILURE",
        "InstanceType": "BASIC",
        "Remark": "",
        "SkuCode": "mqtt_1k",
        "SubscriptionNumLimit": 1,
        "TopicNum": 0,
```

```
"TopicNumLimit": 1,
"TPSLimit": 1,
"Version": "1"
},
{
"ClientNumLimit": 1000,
"CreateTime": 1705391046,
"InstanceId": "mqtt-47ka4rdr",
"InstanceName": "sun-0116",
"InstanceStatus": "RUNNING",
"InstanceType": "BASIC",
"Remark": "",
"SKUCode": "basic_1k",
"SubscriptionNumLimit": 1000,
"TopicNum": 25,
"TopicNumLimit": 25,
"TPSLimit": 1000,
"Version": "1"
}
],
"RequestId": "9424369d-904f-4fa2-b407-0bd5e536e989",
"TotalCount": 10
}
}
```

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: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub Gitee](#)

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	操作失败。

查询MQTT实例详情信息

最近更新时间：2024-04-22 01:28:22

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

查询实例信息

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：DescribeMQTTInstance。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
InstanceId	是	String	实例ID 示例值：rocketmq-xxxx

3. 输出参数

参数名称	类型	描述
InstanceType	String	实例类型， EXPERIMENT 体验版 BASIC 基础版 PRO 专业版 PLATINUM 铂金版 示例值：BASIC
InstanceId	String	实例ID 示例值：mqtt-mwmdg9dm
InstanceName	String	实例名称 示例值：name
TopicNum	Integer	主题数量 示例值：2
TopicNumLimit	Integer	实例最大主题数量 示例值：1000
TpsLimit	Integer	TPS限流值 示例值：1000

参数名称	类型	描述
CreateTime	Integer	创建时间，秒为单位 示例值：1711955420
Remark	String	备注信息 示例值：remark
InstanceStatus	String	实例状态 示例值：RUNNING
SkuCode	String	实例规格 示例值：basic_1k
SubscriptionNumLimit	Integer	订阅数上限 示例值：1000
ClientNumLimit	Integer	客户端数量上限 示例值：1000
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 示例

示例

输入示例

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

{
  "InstanceId": "mqtt-7pnqmkrx"
}
```

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "ClientNumLimit": 1000,
    "CreateTime": 1705562417,
    "InstanceId": "mqtt-7pnqmkrx",
    "InstanceName": "sunjianxiang-1",
    "InstanceStatus": "RUNNING",
    "InstanceType": "BASIC",
    "Remark": "this is remark",
    "RequestId": "2251b9cf-aedd-4457-ac8d-bc705f48ccb8",
```



```

"SkuCode": "basic_1k",
"SubscriptionNumLimit": 1000,
"TopicNum": 1,
"TopicNumLimit": 25,
"TpsLimit": 1000
}
}
    
```

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: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub Gitee](#)

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
ResourceNotFound.Instance	实例不存在。

创建MQTT主题

最近更新时间：2024-04-22 01:28:24

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

创建主题

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：CreateMQTTTopic。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
InstanceId	是	String	实例ID 示例值：rocketmq-xxxx
Topic	是	String	主题 示例值：test
Remark	否	String	备注 示例值：remark

3. 输出参数

参数名称	类型	描述
InstanceId	String	实例ID 示例值：mqtt-1234
Topic	String	主题 示例值：test
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 示例

示例

输入示例

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

{
  "InstanceId": "mqtt-7pnqmkrx",
  "Topic": "topic1"
}
```

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "InstanceId": "mqtt-7pnqmkrx",
    "RequestId": "dff879db-cb6f-4ffb-b820-ea6f391a8da7",
    "Topic": "topic1"
  }
}
```

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: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub Gitee](#)

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	操作失败。
FailedOperation.InstanceNotReady	实例未就绪，请稍后重试。
LimitExceeded.TopicNum	已达到主题数上限。
OperationDenied	操作被拒绝。
ResourceNotFound.Instance	实例不存在。
UnsupportedOperation.ResourceAlreadyExists	资源已存在，请检查后重试。

删除MQTT主题

最近更新时间：2024-04-22 01:28:23

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

删除MQTT主题

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：DeleteMQTTTopic。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
InstanceId	是	String	实例ID 示例值：rocketmq-xxxx
Topic	是	String	主题 示例值：test

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 示例

示例

输入示例

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

```
{
  "InstanceId": "mqtt-rvb5xxrn",
  "Topic": "111"
}
```

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "RequestId": "f3a081b8-dc97-4d62-bb1a-d1aa9fe80da8"
  }
}
```

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: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub Gitee](#)

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	操作失败。
ResourceNotFound.Topic	主题不存在，请检查后重试。

修改MQTT主题属性

最近更新时间：2024-03-12 01:58:22

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

修改主题属性

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：ModifyMQTTTopic。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
InstanceId	是	String	实例ID 示例值：rocketmq-xxxx
Topic	是	String	主题 示例值：test
Remark	否	String	备注信息 示例值：test

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 示例

示例

输入示例

```
POST / HTTP/1.1
Host: trocket.tencentcloudapi.com
Content-Type: application/json
```

```
X-TC-Action: ModifyMQTTTopic
```

```
<公共请求参数>
```

```
{
  "InstanceId": "mqtt-rvb5xxrn",
  "Topic": "222",
  "Remark": "444"
}
```

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "RequestId": "29a02c3b-66ee-4442-acb7-3545b8e044b7"
  }
}
```

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: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub Gitee](#)

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
-----	----

错误码	描述
FailedOperation	操作失败。

查询MQTT主题列表

最近更新时间：2024-04-22 01:28:21

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

获取主题列表，Filter参数使用说明如下：

1. TopicName，主题名称模糊搜索
2. TopicType，主题类型查询，支持多选，可选值：Normal,Order,Transaction,DelayScheduled

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：DescribeMQTTTopicList。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
InstanceId	是	String	实例ID 示例值：rocketmq-xxxx
Filters.N	否	Array of Filter	查询条件列表
Offset	否	Integer	查询起始位置 示例值：0
Limit	否	Integer	查询结果限制数量 示例值：10

3. 输出参数

参数名称	类型	描述
TotalCount	Integer	查询总数 注意：此字段可能返回 null，表示取不到有效值。 示例值：10
Data	Array of MQTTTopicItem	主题列表 示例值：[{ "InstanceId": "mqtt-mwmdg9dm", "Topic": "2222", "Remark": "" }, { "InstanceId": "mqtt-mwmdg9dm", "Topic": "home", "Remark": "" }]
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 示例

示例

输入示例

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

{
  "InstanceId": "mqtt-47ka4rdr",
  "Offset": 0,
  "Limit": 20
}
```

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "Data": [
      {
        "InstanceId": "mqtt-47ka4rdr",
        "Remark": "",
        "Topic": "1111"
      },
      {
        "InstanceId": "mqtt-47ka4rdr",
        "Remark": "",
        "Topic": "topic24"
      },
      {
        "InstanceId": "mqtt-47ka4rdr",
        "Remark": "",
        "Topic": "topic23"
      },
      {
        "InstanceId": "mqtt-47ka4rdr",
        "Remark": "",
        "Topic": "topic22"
      },
      {
        "InstanceId": "mqtt-47ka4rdr",
        "Remark": "",
        "Topic": "topic21"
      }
    ]
  }
}
```

```
},
{
  "Instanceld": "mqtt-47ka4rdr",
  "Remark": "",
  "Topic": "topic20"
},
{
  "Instanceld": "mqtt-47ka4rdr",
  "Remark": "",
  "Topic": "topic19"
},
{
  "Instanceld": "mqtt-47ka4rdr",
  "Remark": "",
  "Topic": "topic18"
},
{
  "Instanceld": "mqtt-47ka4rdr",
  "Remark": "",
  "Topic": "topic17"
},
{
  "Instanceld": "mqtt-47ka4rdr",
  "Remark": "",
  "Topic": "topic16"
},
{
  "Instanceld": "mqtt-47ka4rdr",
  "Remark": "",
  "Topic": "topic15"
},
{
  "Instanceld": "mqtt-47ka4rdr",
  "Remark": "",
  "Topic": "topic14"
},
{
  "Instanceld": "mqtt-47ka4rdr",
  "Remark": "",
  "Topic": "topic13"
},
{
  "Instanceld": "mqtt-47ka4rdr",
  "Remark": "",
  "Topic": "topic12"
},
{
  "Instanceld": "mqtt-47ka4rdr",
  "Remark": "",
  "Topic": "topic11"
}
```

```
},
{
  "InstanceId": "mqtt-47ka4rdr",
  "Remark": "",
  "Topic": "topic10"
},
{
  "InstanceId": "mqtt-47ka4rdr",
  "Remark": "",
  "Topic": "topic9"
},
{
  "InstanceId": "mqtt-47ka4rdr",
  "Remark": "",
  "Topic": "topic8"
},
{
  "InstanceId": "mqtt-47ka4rdr",
  "Remark": "",
  "Topic": "topic7"
},
{
  "InstanceId": "mqtt-47ka4rdr",
  "Remark": "",
  "Topic": "topic6"
}
],
"RequestId": "d9042e88-c70e-4158-958c-8bb57d6c809f",
"TotalCount": 25
}
```

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: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub Gitee](#)

- Tencent Cloud SDK 3.0 for .NET: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub](#) [Gitee](#)

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
ResourceNotFound.Instance	实例不存在。

查询MQTT消息列表

最近更新时间：2024-04-22 01:28:22

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

查询消息列表，如查询死信，请设置ConsumerGroup参数

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：DescribeMQTTMessageList。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
InstanceId	是	String	实例ID 示例值：rocketmq-xxxx
Topic	是	String	主题 示例值：test
StartTime	是	Integer	开始时间 示例值：1712025623202
EndTime	是	Integer	结束时间 示例值：1712112023202
TaskRequestId	是	String	请求任务id 示例值：dcfefe36-829d-4dd8-a436-e622c6ecab8f
Offset	否	Integer	查询起始位置 示例值：0
Limit	否	Integer	查询结果限制数量 示例值：10

3. 输出参数

参数名称	类型	描述
TotalCount	Integer	查询总数 注意：此字段可能返回 null，表示取不到有效值。 示例值：10

参数名称	类型	描述
Data	Array of MQTTMessageItem	消息记录列表 注意：此字段可能返回 null，表示取不到有效值。 示例值：[{"MsgId": "0B8D67D3003505C29BFD077EC81A0005", "SubTopic": "/humidity/room1/", "Qos": "2", "Tags": "MQTT_COMMON", "Keys": null, "ProducerAddr": "11.176.16.153:26866", "ProduceTime": "2024-04-02 10:55:49", "DeadLetterResendTimes": null, "DeadLetterResendSuccessTimes": null}, {"MsgId": "0B8D67D3003505C29BFD077EC65B0002", "SubTopic": "/humidity/room0/", "Qos": "2", "Tags": "MQTT_COMMON", "Keys": null, "ProducerAddr": "11.176.16.153:26866", "ProduceTime": "2024-04-02 10:55:48", "DeadLetterResendTimes": null, "DeadLetterResendSuccessTimes": null}]
TaskRequestId	String	请求任务id 示例值：dcfef36-829d-4dd8-a436-e622c6ecab8f
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 示例

示例

输入示例

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

{
  "InstanceId": "mqtt-47ka4rdr",
  "Topic": "topic24",
  "Offset": 0,
  "Limit": 20,
  "StartTime": 1705373755266,
  "EndTime": 1705632955266,
  "TaskRequestId": ""
}
```

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "Data": [
      {
        "DeadLetterResendSuccessTimes": null,
```



```
"DeadLetterResendTimes": null,
"Keys": null,
"MsgId": "0B8D67F0002D6ACBCFC0557BCEBA0001",
"ProduceTime": "2024-01-17 14:22:57",
"ProducerAddr": "11.176.16.153:51134",
"Qos": "1",
"SubTopic": "/1/",
"Tags": "MQTT_COMMON"
},
{
"DeadLetterResendSuccessTimes": null,
"DeadLetterResendTimes": null,
"Keys": null,
"MsgId": "0B8D67F0002D6ACBCFC0557B1A980000",
"ProduceTime": "2024-01-17 14:22:11",
"ProducerAddr": "11.176.16.153:44044",
"Qos": "0",
"SubTopic": "",
"Tags": "MQTT_COMMON"
}
],
"RequestId": "116cd3f5-668f-42fe-8572-fcd5d3a31377",
"TaskRequestId": "e3d14393-4701-4f62-8a7d-b52bd59b5ace",
"TotalCount": 2
}
}
```

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: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub Gitee](#)

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
ResourceNotFound.Instance	实例不存在。

添加MQTT角色

最近更新时间：2024-04-22 01:28:24

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

添加mqtt角色

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：CreateMQTTUser。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
InstanceId	是	String	实例ID 示例值：rocketmq-xxxx
Remark	是	String	备注 示例值：remark
PermWrite	是	Boolean	是否开启生产权限 示例值：true
PermRead	是	Boolean	是否开启消费权限 示例值：false
Username	是	String	用户名 示例值：user1
Password	否	String	密码，该字段为空时候则后端会默认生成 示例值：pwd

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 示例

示例

输入示例

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

{
  "InstanceId": "mqtt-7pnqmkrx",
  "Remark": "33333",
  "PermWrite": true,
  "PermRead": true,
  "Username": "123",
  "Password": "456"
}
```

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "RequestId": "25cb77a7-2add-4086-8fd2-a05b5bd34776"
  }
}
```

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: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub Gitee](#)

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	操作失败。
FailedOperation.InstanceNotReady	实例未就绪，请稍后重试。
OperationDenied	操作被拒绝。
ResourceInUse	资源被占用。
ResourceNotFound.Instance	实例不存在。
UnsupportedOperation.ResourceAlreadyExists	资源已存在，请检查后重试。

查询MQTT用户列表

最近更新时间：2024-04-22 01:28:21

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

查询用户列表，Filter参数使用说明如下：

1. Username，用户名称模糊搜索

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：DescribeMQTTUserList。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
InstanceId	是	String	实例ID 示例值：rocketmq-xxxx
Filters.N	否	Array of Filter	查询条件列表
Offset	否	Integer	查询起始位置 示例值：0
Limit	否	Integer	查询结果限制数量 示例值：10

3. 输出参数

参数名称	类型	描述
TotalCount	Integer	查询总数 注意：此字段可能返回 null，表示取不到有效值。 示例值：10
Data	Array of MQTTUserItem	角色信息列表 示例值：[{"CreatedTime": 1705392159000, "ModifiedTime": 1705392159000, "Password": "sunjianxiang", "PermRead": true, "PermWrite": true, "Remark": "", "Username": "user1"}]
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 示例

示例

输入示例

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

{
  "InstanceId": "mqtt-47ka4rdr",
  "Offset": 0,
  "Limit": 10
}
```

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "Data": [
      {
        "CreatedTime": 1705392159000,
        "ModifiedTime": 1705392159000,
        "Password": "sunjianxiong",
        "PermRead": true,
        "PermWrite": true,
        "Remark": "",
        "Username": "user1"
      }
    ],
    "RequestId": "8e86117c-a56c-4598-9e1f-4a039eed6e3a",
    "TotalCount": 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: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub](#) [Gitee](#)

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

该接口暂无业务逻辑相关的错误码，其他错误码详见 [公共错误码](#)。

修改MQTT角色

最近更新时间：2024-04-22 01:28:20

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

修改MQTT角色

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：ModifyMQTTUser。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
InstanceId	是	String	实例ID 示例值：rocketmq-xxxx
Username	是	String	用户名 示例值：user1
PermRead	是	Boolean	是否开启消费 示例值：true
PermWrite	是	Boolean	是否开启生产 示例值：false
Remark	否	String	备注 示例值：remark

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 示例

示例

输入示例

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

{
  "InstanceId": "mqtt-7pnqmkrx",
  "Username": "222",
  "Remark": "3333",
  "PermRead": true,
  "PermWrite": true
}
```

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "RequestId": "85d3ae4e-ba25-47cc-8c17-7ad837b877ba"
  }
}
```

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: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub](#) [Gitee](#)

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	操作失败。

删除MQTT角色

最近更新时间：2024-04-22 01:28:23

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

删除MQTT访问用户

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：DeleteMQTTUser。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
InstanceId	是	String	实例ID 示例值：rocketmq-xxxx
Username	是	String	用户名 示例值：user1

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 示例

示例

输入示例

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

```
{
  "InstanceId": "mqtt-7pnqmkrx",
  "Username": "123"
}
```

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "RequestId": "74625f9b-ce92-43d5-ba0b-9a85ddb99041"
  }
}
```

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: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub Gitee](#)

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	操作失败。
ResourceNotFound.Instance	实例不存在。

错误码	描述
ResourceNotFound.Role	角色不存在，请检查后重试。

获取MQTT产品售卖规格

最近更新时间：2024-04-22 01:28:22

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

获取产品售卖规格

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：DescribeMQTTProductSKUList。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。

3. 输出参数

参数名称	类型	描述
TotalCount	Integer	查询总数 注意：此字段可能返回 null，表示取不到有效值。 示例值：10
MQTTProductSkuList	Array of MQTTProductSkulItem	mqtt商品配置信息 注意：此字段可能返回 null，表示取不到有效值。 示例值：[[{"ClientNumLimit": 1, "InstanceType": "BASIC", "OnSale": true, "ProxySpecCore": 2, "ProxySpecCount": 1, "ProxySpecMemory": 4, "SkuCode": "mqtt_1k", "SubscriptionNumLimit": 1, "TopicNumLimit": 1, "TpsLimit": 1}, {"ClientNumLimit": 1000, "InstanceType": "BASIC", "OnSale": true, "ProxySpecCore": 1, "ProxySpecCount": 2, "ProxySpecMemory": 2, "SkuCode": "basic_1k", "SubscriptionNumLimit": 1000, "TopicNumLimit": 25, "TpsLimit": 1000}]]
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 示例

示例

输入示例

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

{}
```

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "MQTTProductSkuList": [
      {
        "ClientNumLimit": 1,
        "InstanceType": "BASIC",
        "OnSale": true,
        "ProxySpecCore": 2,
        "ProxySpecCount": 1,
        "ProxySpecMemory": 4,
        "SkuCode": "mqtt_1k",
        "SubscriptionNumLimit": 1,
        "TopicNumLimit": 1,
        "TpsLimit": 1
      },
      {
        "ClientNumLimit": 1000,
        "InstanceType": "BASIC",
        "OnSale": true,
        "ProxySpecCore": 1,
        "ProxySpecCount": 2,
        "ProxySpecMemory": 2,
        "SkuCode": "basic_1k",
        "SubscriptionNumLimit": 1000,
        "TopicNumLimit": 25,
        "TpsLimit": 1000
      }
    ],
    "RequestId": "2be05982-b125-4259-b965-c487949cd896",
    "TotalCount": 2
  }
}
```

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: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub Gitee](#)

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

该接口暂无业务逻辑相关的错误码，其他错误码详见 [公共错误码](#)。

购买MQTT实例

最近更新时间：2024-04-22 01:28:24

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

购买新的MQTT实例

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：CreateMQTTInstance。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
InstanceType	是	String	实例类型， EXPERIMENT 体验版 BASIC 基础版 PRO 专业版 PLATINUM 铂金版 示例值：BASIC
Name	是	String	实例名称 示例值：abcd

参数名称	必选	类型	描述
Skucode	是	String	商品规格，可用规格如下： experiment_500, basic_1k, basic_2k, basic_4k, basic_6k, pro_4k, pro_6k, pro_1w, pro_2w, pro_3w, pro_4w, pro_5w, platinum_6k, platinum_1w, platinum_2w, platinum_4w, platinum_10w, platinum_15w, platinum_20w, platinum_40w, platinum_60w, platinum_100w 示例值：basic_1k
Remark	否	String	备注信息 示例值：remark
TagList.N	否	Array of Tag	标签列表 示例值：[]
VpcList.N	否	Array of VpcInfo	实例绑定的VPC信息 示例值：[{"VpcId": "vpc-9a5wnirv", "SubnetId": "subnet-0tw6tlfi"}]
EnablePublic	否	Boolean	是否开启公网 示例值：true
Bandwidth	否	Integer	公网带宽（单位：兆） 示例值：1
IpRules.N	否	Array of IpRule	公网访问白名单 示例值：[{"Ip": "1.1.1.1", "Allow": true, "Remark": "11111111"}]
RenewFlag	否	Integer	是否自动续费（0：不自动续费；1：自动续费） 示例值：1
TimeSpan	否	Integer	购买时长（单位：月） 示例值：1

3. 输出参数

参数名称	类型	描述
InstanceId	String	实例ID 示例值：rmq-xxxx
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 示例

示例

输入示例

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

{
  "InstanceType": "BASIC",
  "Name": "sunjianxiong-1",
  "SkuCode": "basic_1k",
  "Remark": "this is remark",
  "VpcList": [
    {
      "VpcId": "vpc-9a5wnirv",
      "SubnetId": "subnet-0tw6tlfi"
    }
  ],
  "IpRules": [
    {
      "Ip": "1.1.1.1",
      "Allow": true,
      "Remark": "111111111"
    }
  ]
}
```

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "InstanceId": "mqtt-7pnqmkrx",
    "RequestId": "02107c9d-cbea-418c-bafc-740300fdefad"
  }
}
```

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: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub Gitee](#)

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	操作失败。

删除MQTT实例

最近更新时间：2024-04-22 01:28:23

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

删除MQTT实例

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：DeleteMQTTInstance。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
InstanceId	是	String	实例ID 示例值：rocketmq-xxxx

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 示例

示例

输入示例

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

{
```

```
"InstanceId": "mqtt-wjgxr8jg"
}
```

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "RequestId": "84afeb76-9175-47ec-a6f1-8d0af0c0aae6"
  }
}
```

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: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub Gitee](#)

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	操作失败。
ResourceInUse	资源被占用。
ResourceNotFound.Instance	实例不存在。

修改MQTT实例属性

最近更新时间：2024-04-22 01:28:21

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

修改实例属性

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：ModifyMQTTInstance。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
InstanceId	是	String	实例ID 示例值：rocketmq-xxxx
Name	否	String	实例名称 示例值：instance1
Remark	否	String	备注信息 示例值：remark

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 示例

示例

输入示例

```
POST / HTTP/1.1
Host: trocket.tencentcloudapi.com
Content-Type: application/json
```


X-TC-Action: ModifyMQTTInstance

<公共请求参数>

```
{
  "InstanceId": "mqtt-47ka4rdr",
  "Name": "sjx0119",
  "Remark": "555"
}
```

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "RequestId": "9955522b-ac27-46c1-8a16-3ed6f98f9c5c"
  }
}
```

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: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub Gitee](#)

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
-----	----

错误码	描述
FailedOperation	操作失败。
FailedOperation.InstanceNotReady	实例未就绪，请稍后重试。
ResourceUnavailable	资源不可用。

查询MQTT主题详情

最近更新时间：2024-04-22 01:28:21

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

查询mqtt主题详情

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：DescribeMQTTTopic。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
InstanceId	是	String	实例ID 示例值：rocketmq-xxxx
Topic	是	String	主题 示例值：test

3. 输出参数

参数名称	类型	描述
InstanceId	String	实例ID 示例值：mqtt-mwmdg9dm
Topic	String	主题名称 示例值：topic1
Remark	String	备注 示例值：remark
CreateTime	Integer	创建时间，秒为单位 示例值：1712049671
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 示例

示例

输入示例

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

{
  "InstanceId": "mqtt-7pnqmkrx",
  "Topic": "topic1"
}
```

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "CreateTime": 1705567642,
    "InstanceId": "mqtt-7pnqmkrx",
    "Remark": "",
    "RequestId": "acf7faf2-b2a9-431c-96b9-f589262da8f7",
    "Topic": "topic1"
  }
}
```

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: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub Gitee](#)

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
ResourceNotFound.Instance	实例不存在。
ResourceNotFound.Topic	主题不存在，请检查后重试。

更新MQTT集群证书证书

最近更新时间：2024-04-22 01:28:20

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

更新MQTT集群绑定证书

参数传空，则为删除证书

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：ModifyMQTTInstanceCertBinding。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
InstanceId	是	String	实例ID 示例值：rocketmq-xxxx
SSLServerCertId	是	String	服务端证书id 示例值：serverCertId
SSLCaCertId	是	String	CA证书id 示例值：caId

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 示例

示例

输入示例

```
POST / HTTP/1.1
Host: trocket.tencentcloudapi.com
```

```
Content-Type: application/json
X-TC-Action: ModifyMQTTInstanceCertBinding
<公共请求参数>
```

```
{
  "InstanceId": "mqtt-rvb5xxrn",
  "SSLServerCertId": " ",
  "SSLCaCertId": " "
}
```

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "RequestId": "1d2d027f-1701-4bc6-8a2a-cede24d9c9ed"
  }
}
```

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: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub Gitee](#)

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

该接口暂无业务逻辑相关的错误码，其他错误码详见 [公共错误码](#)。

查询MQTT集群证书列表

最近更新时间：2024-04-22 01:28:22

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

查询MQTT集群证书列表

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：DescribeMQTTInstanceCert。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
InstanceId	是	String	实例ID 示例值：rocketmq-xxxx

3. 输出参数

参数名称	类型	描述
InstanceId	String	集群id 注意：此字段可能返回 null，表示取不到有效值。 示例值：mqtt-mwmdg9dm
SSLServerCertId	String	服务端证书id 注意：此字段可能返回 null，表示取不到有效值。 示例值：certified
SSLCaCertId	String	CA证书id 注意：此字段可能返回 null，表示取不到有效值。 示例值：lacertid
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 示例

示例

输入示例

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

{
  "InstanceId": "mqtt-7pnqmkrx"
}
```

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "InstanceId": null,
    "RequestId": "a07a950a-43ea-4c13-a812-7ba4337d186a",
    "SSLCaCertId": null,
    "SSLServerCertId": null
  }
}
```

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: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub Gitee](#)

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

该接口暂无业务逻辑相关的错误码，其他错误码详见 [公共错误码](#)。

查询 MQTT 客户端详情

最近更新时间：2024-04-22 01:28:23

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

查询 MQTT 客户端详情

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：DescribeMQTTClient。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
InstanceId	是	String	实例ID 示例值：rocketmq-xxxx
ClientId	是	String	客户端详情 示例值：client1

3. 输出参数

参数名称	类型	描述
ClientId	String	客户端唯一标识 示例值：myclient001
ClientAddress	String	客户端网络地址 示例值：127.0.0.1:10001
ProtocolVersion	Integer	MQTT 协议版本，4 表示 MQTT 3.1.1 示例值：4
Keepalive	Integer	保持连接时间，单位：秒 示例值：60
ConnectionStatus	String	连接状态，CONNECTED 已连接，DISCONNECTED 未连接 示例值：CONNECTED
CreateTime	Integer	客户端创建时间 示例值：1701396000000

参数名称	类型	描述
ConnectTime	Integer	上次建立连接时间 示例值：1701396000000
DisconnectTime	Integer	上次断开连接时间，仅对持久会话（cleanSession=false）并且客户端当前未连接时有意义 示例值：1701403200000
MQTTClientSubscriptions	Array of MQTTClientSubscription	客户端的订阅列表 示例值：[{"Qos": 1, "TopicFilter": "home/sun/+/"}, {"Qos": 1, "TopicFilter": "home/temp/+/"}, {"Qos": 0, "TopicFilter": "home/humidity/+/"}, {"Qos": 2, "TopicFilter": "home/person/+/"}]
Inbound	StatisticsReport	服务端到客户端的流量统计 示例值：{"Bytes": 16, "PacketQos0": 4, "PacketQos1": 0, "PacketQos2": 0, "PacketTotal": 4}
OutBound	StatisticsReport	客户端到服务端的流量统计 示例值：{"Bytes": 16, "PacketQos0": 4, "PacketQos1": 0, "PacketQos2": 0, "PacketTotal": 4}
CleanSession	Boolean	cleansession标志 示例值：true
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 样例

样例

输入示例

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

{
  "InstanceId": "mqtt-2vnk55xv",
  "ClientId": "sub_client2"
}
```

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "CleanSession": true,
```

```

"ClientAddress": "11.176.16.153:23354",
"ClientId": "sub_client2",
"ConnectTime": 1711444795304,
"ConnectionStatus": "CONNECTED",
"CreateTime": 1711443908331,
"DisconnectTime": 0,
"Inbound": {
  "Bytes": 40,
  "Items": [
    {
      "Count": 1,
      "MessageType": "SUBACK",
      "Qos": 0
    },
    {
      "Count": 1,
      "MessageType": "CONNACK",
      "Qos": 0
    },
    {
      "Count": 14,
      "MessageType": "PINGRESP",
      "Qos": 0
    }
  ]
},
"Keepalive": 60,
"MQTTClientSubscriptions": [
  {
    "Qos": 1,
    "TopicFilter": "home/sun/+"
  },
  {
    "Qos": 1,
    "TopicFilter": "home/temp/+"
  },
  {
    "Qos": 0,
    "TopicFilter": "home/humidity/+"
  },
  {
    "Qos": 2,
    "TopicFilter": "home/person/+"
  }
],
"OutBound": {
  "Bytes": 134,
  "Items": [
    {
      "Count": 1,

```

```

"MessageType": "CONNECT",
"Qos": 0
},
{
"Count": 1,
"MessageType": "SUBSCRIBE",
"Qos": 1
},
{
"Count": 14,
"MessageType": "PINGREQ",
"Qos": 0
}
]
},
"ProtocolVersion": 4,
"RequestId": "b59aa63c-6cc6-418a-92c7-c205cb30d4cd"
}
}
    
```

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: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub Gitee](#)

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
-----	----

错误码	描述
ResourceNotFound.Client	客户端不存在。

为MQTT实例创建公网接入点

最近更新时间：2024-04-22 01:28:24

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

为MQTT实例创建公网接入点

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：CreateMQTTInsPublicEndpoint。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
InstanceId	是	String	实例ID 示例值：rocketmq-xxxx
Bandwidth	是	Integer	带宽 示例值：2
Rules.N	否	Array of PublicAccessRule	公网访问规则 示例值：[{"IpRule": "1.1.1.1", "Allow": true, "Remark": "remark"}]

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 示例

示例

输入示例

```
POST / HTTP/1.1
Host: trocket.tencentcloudapi.com
Content-Type: application/json
```



```
X-TC-Action: CreateMQTTInsPublicEndpoint
```

```
<公共请求参数>
```

```
{
  "InstanceId": "mqtt-rvb5xxrn",
  "Bandwidth": 2,
  "Rules": [
    {
      "IpRule": "1.1.1.1",
      "Allow": true,
      "Remark": "remark"
    }
  ]
}
```

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "RequestId": "416510a6-f614-487d-bbca-e7fdbf72fc29"
  }
}
```

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: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub Gitee](#)

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

该接口暂无业务逻辑相关的错误码，其他错误码详见 [公共错误码](#)。

删除MQTT实例的公网接入点

最近更新时间：2024-03-12 01:58:25

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

删除MQTT实例的公网接入点

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：DeleteMQTTInsPublicEndpoint。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
InstanceId	是	String	实例ID 示例值：rocketmq-xxxx

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 示例

示例

输入示例

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

```
{
```

```
"InstanceId": "mqtt-rvb5xxrn"
}
```

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "RequestId": "de89c7dd-7c3e-4546-a2ca-91c5baa0415f"
  }
}
```

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: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub Gitee](#)

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

该接口暂无业务逻辑相关的错误码，其他错误码详见 [公共错误码](#)。

查询MQTT实例公网接入点

最近更新时间：2024-04-22 01:28:22

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

查询MQTT实例公网接入点

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：DescribeMQTTInsPublicEndpoints。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
InstanceId	是	String	实例ID 示例值：rocketmq-xxxx

3. 输出参数

参数名称	类型	描述
Endpoints	Array of MQTTEndpointItem	接入点 示例值：[{ "Type": "mqtt-tcp", "Url": "mqtt-abc1234-gz.mqtt.public.tencenttdmq.com:1883", "VpcId": null, "SubnetId": null, "Host": "mqtt-abc1234-gz.mqtt.public.tencenttdmq.com", "Port": 1883 }, { "Type": "mqtt-ws", "Url": "mqtt-abc1234-gz.mqtt.public.tencenttdmq.com:8888", "VpcId": null, "SubnetId": null, "Host": "mqtt-abc1234-gz.mqtt.public.tencenttdmq.com", "Port": 8888 }]
InstanceId	String	实例id
Bandwidth	Integer	带宽 示例值：2
Rules	Array of PublicAccessRule	公网访问规则 示例值：[{ "IpRule": "127.0.0.1", "Allow": true, "Remark": null }]

参数名称	类型	描述
Status	String	公网状态： NORMAL-正常 CLOSING-关闭中 MODIFYING-修改中 CREATING-开启中 CLOSE-关闭 示例值：NORMAL
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 示例

示例

输入示例

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

{
  "InstanceId": "mqtt-gn8qoq3z"
}
```

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "Bandwidth": 2,
    "Endpoints": [
      {
        "Host": "mqtt-gn8qoq3z-cd.mqtt.public.tencentddmq.com",
        "Port": 22,
        "SubnetId": null,
        "Type": "mqtt-ws",
        "Url": null,
        "VpcId": null
      },
      {
        "Host": "mqtt-gn8qoq3z-cd.mqtt.public.tencentddmq.com",
        "Port": 11,
        "SubnetId": null,
        "Type": "mqtt-tcp",
      }
    ]
  }
}
```

```
"Url": null,
"VpcId": null
},
{
"Host": "mqtt-gn8qoq3z-cd.mqtt.public.tencenttdmq.com",
"Port": 33,
"SubnetId": null,
"Type": "mqtt-tls",
"Url": null,
"VpcId": null
}
],
"InstanceId": "mqtt-gn8qoq3z",
"RequestId": "3891d712-94f7-4d03-9420-fae4fa3a61ea",
"Rules": [
{
"Allow": true,
"IpRule": "0.0.0.0/0",
"Remark": ""
}
],
"Status": "NORMAL"
}
}
```

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: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub Gitee](#)

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

该接口暂无业务逻辑相关的错误码，其他错误码详见 [公共错误码](#)。

查询MQTT实例VPC接入点

最近更新时间：2024-04-22 01:28:22

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

查询MQTT实例公网接入点

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：DescribeMQTTInsVPCEndpoints。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
InstanceId	是	String	实例ID 示例值：rocketmq-xxxx

3. 输出参数

参数名称	类型	描述
Endpoints	Array of MQTTEndpointItem	接入点 示例值：[{"Host": "mqtt-47ka4rdr-cd-qcloud.mqtt.tencenttdmq.com", "Port": 1883, "SubnetId": null, "Type": "mqtt-tcp", "Url": "mqtt-47ka4rdr-cd-qcloud.mqtt.tencenttdmq.com:1883", "VpcId": null}, {"Host": "mqtt-47ka4rdr-cd-qcloud.mqtt.tencenttdmq.com", "Port": 8888, "SubnetId": null, "Type": "mqtt-ws", "Url": "mqtt-47ka4rdr-cd-qcloud.mqtt.tencenttdmq.com:8888", "VpcId": null}]
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 样例

样例

输入示例

```
POST / HTTP/1.1
Host: trocket.tencentcloudapi.com
```

```
Content-Type: application/json
X-TC-Action: DescribeMQTTInsVPCEndpoints
```

<公共请求参数>

```
{
  "InstanceId": "mqtt-25zqb75a"
}
```

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "Endpoints": [
      {
        "Host": "mqtt-25zqb75a-qy.mqtt.qcloud.tencenttdmq.com",
        "Ip": "10.0.0.25",
        "Port": 1883,
        "SubnetId": "subnet-fr9ez60m",
        "Type": "mqtt-tcp",
        "Url": "mqtt-25zqb75a-qy.mqtt.qcloud.tencenttdmq.com:1883",
        "VpcId": "vpc-35wk4c6h"
      },
      {
        "Host": "mqtt-25zqb75a-qy.mqtt.qcloud.tencenttdmq.com",
        "Ip": "10.0.0.25",
        "Port": 8888,
        "SubnetId": "subnet-fr9ez60m",
        "Type": "mqtt-ws",
        "Url": "mqtt-25zqb75a-qy.mqtt.qcloud.tencenttdmq.com:8888",
        "VpcId": "vpc-35wk4c6h"
      }
    ],
    "RequestId": "4313484d-47c1-4fc8-b402-f997d2ba7256"
  }
}
```

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: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub](#) [Gitee](#)

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

该接口暂无业务逻辑相关的错误码，其他错误码详见 [公共错误码](#)。

更新MQTT实例公网接入点

最近更新时间：2024-04-22 01:28:21

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

更新MQTT实例公网接入点

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：ModifyMQTTInsPublicEndpoint。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
InstanceId	是	String	实例ID 示例值：rocketmq-xxxx
Bandwidth	是	Integer	带宽 示例值：2
Rules.N	是	Array of PublicAccessRule	公网访问规则 示例值：[{"IpRule": "2.2.2.2", "Allow": true, "Remark": "555"}]

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 示例

示例

输入示例

```
POST / HTTP/1.1
Host: trocket.tencentcloudapi.com
Content-Type: application/json
```

```
X-TC-Action: ModifyMQTTInsPublicEndpoint
```

```
<公共请求参数>
```

```
{
  "InstanceId": "mqtt-47ka4rdr",
  "Bandwidth": 2,
  "Rules": [
    {
      "IpRule": "2.2.2.2",
      "Allow": true,
      "Remark": "555"
    }
  ]
}
```

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "RequestId": "02eb3529-3440-44e9-b2c8-1e65c7017c09"
  }
}
```

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: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub Gitee](#)

命令行工具

- [Tencent Cloud CLI 3.0](#)

6. 错误码

该接口暂无业务逻辑相关的错误码，其他错误码详见 [公共错误码](#)。

查询MQTT消息

最近更新时间：2024-04-22 01:28:20

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

查询MQTT消息详情

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：DescribeMQTTMessage。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
InstanceId	是	String	实例ID 示例值：rocketmq-xxxx
Topic	是	String	主题 示例值：test
MsgId	是	String	消息ID 示例值：0B8D67D3003505C29BFD077EC65B0002

3. 输出参数

参数名称	类型	描述
Body	String	消息体 示例值：body

参数名称	类型	描述
Properties	Array of CustomMapEntry	详情参数 示例值: [{"Key": "TRACE_ON", "Value": "true"}, {"Key": "originMqttTopic", "Value": "mqtt-mwmdg9dm%home/humidity/room0"}, {"Key": "INNER_MULTI_DISPATCH", "Value": "%LMQ%mqtt-mwmdg9dm%home%humidity%+%,%LMQ%mqtt-mwmdg9dm%home%humidity%room0%"}, {"Key": "IS_EMPTY_MSG", "Value": "false"}, {"Key": "INNER_MULTI_QUEUE_OFFSET", "Value": "30,40"}, {"Key": "retryTimes", "Value": "0"}, {"Key": "extData", "Value": "{"qosLevel": "2"}"}, {"Key": "MSG_REGION", "Value": "cd"}, {"Key": "qosLevel", "Value": "2"}, {"Key": "UNIQ_KEY", "Value": "0B8D67D3003505C29BFD077EC65B0002"}, {"Key": "CLUSTER", "Value": "rmqbk-mqtt"}, {"Key": "TAGS", "Value": "MQTT_COMMON"}, {"Key": "_CLIENTHOST", "Value": "11.176.16.153:26866"}]
ProduceTime	String	生产时间 示例值: 2024-04-02 10:55:48
MessageId	String	消息ID 注意: 此字段可能返回 null, 表示取不到有效值。 示例值: 0B8D67D3003505C29BFD077EC65B0002
ProducerAddr	String	生产者地址 示例值: 127.0.0.1:8888
ShowTopicName	String	Topic 示例值: topic1
RequestId	String	唯一请求 ID, 由服务端生成, 每次请求都会返回 (若请求因其他原因未能抵达服务端, 则该次请求不会获得 RequestId)。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 示例

示例

输入示例

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

```
{
  "InstanceId": "mqtt-47ka4rdr",
  "Topic": "topic24",
  "MsgId": "0B8D67F0002D6ACBCFC0557BCEBA0001"
}
```

输出示例

```
{
  "Error": null,
```



```
"RequestId": null,
"Response": {
  "Body": "Hello MQTT111111",
  "MessageId": "0B8D67F0002D6ACBCFC0557BCEBA0001",
  "ProduceTime": "2024-01-17 14:22:57",
  "ProducerAddr": "11.176.16.153:51134",
  "Properties": [
    {
      "Key": "TRACE_ON",
      "Value": "true"
    },
    {
      "Key": "originMqttTopic",
      "Value": "mqtt-47ka4rdr%topic24/1"
    },
    {
      "Key": "INNER_MULTI_DISPATCH",
      "Value": "%LMQ%mqtt-47ka4rdr%topic24%1%"
    },
    {
      "Key": "IS_EMPTY_MSG",
      "Value": "false"
    },
    {
      "Key": "INNER_MULTI_QUEUE_OFFSET",
      "Value": "0"
    },
    {
      "Key": "retryTimes",
      "Value": "0"
    },
    {
      "Key": "extData",
      "Value": "{\"qosLevel\": \"1\"}"
    },
    {
      "Key": "MSG_REGION",
      "Value": "cd"
    },
    {
      "Key": "qosLevel",
      "Value": "1"
    },
    {
      "Key": "UNIQ_KEY",
      "Value": "0B8D67F0002D6ACBCFC0557BCEBA0001"
    },
    {
      "Key": "CLUSTER",
      "Value": "rmqbk-mqtt"
```

```
},
{
  "Key": "TAGS",
  "Value": "MQTT_COMMON"
},
{
  "Key": "__CLIENT_HOST",
  "Value": "11.176.16.153:51134"
}
],
"RequestId": "e658afc8-e2c0-463e-bf90-3103d881d59f",
"ShowTopicName": "topic24"
}
```

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: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub Gitee](#)

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
ResourceNotFound.Instance	实例不存在。
ResourceNotFound.Message	消息不存在。

集群相关接口

购买实例

最近更新时间：2024-03-12 01:58:21

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

购买新实例

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：CreateInstance。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
InstanceType	是	String	实例类型， EXPERIMENT 体验版 BASIC 基础版 PRO 专业版 PLATINUM 铂金版 示例值：BASIC
Name	是	String	实例名称 示例值：abcd

参数名称	必选	类型	描述
Skucode	是	String	商品规格，可用规格如下： experiment_500, basic_1k, basic_2k, basic_4k, basic_6k, pro_4k, pro_6k, pro_1w, pro_2w, pro_3w, pro_4w, pro_5w, platinum_6k, platinum_1w, platinum_2w, platinum_4w, platinum_10w, platinum_15w, platinum_20w, platinum_40w, platinum_60w, platinum_100w
Remark	否	String	备注信息
TagList.N	否	Array of Tag	标签列表
VpcList.N	否	Array of VpcInfo	实例绑定的VPC信息
EnablePublic	否	Boolean	是否开启公网
Bandwidth	否	Integer	公网带宽（单位：兆）
IpRules.N	否	Array of IpRule	公网访问白名单
MessageRetention	否	Integer	消息保留时长（单位：小时）
PayMode	否	Integer	付费模式（0：后付费；1：预付费）
RenewFlag	否	Integer	是否自动续费（0：不自动续费；1：自动续费）
TimeSpan	否	Integer	购买时长（单位：月）
MaxTopicNum	否	Integer	最大可创建主题数 示例值：50

3. 输出参数

参数名称	类型	描述
InstanceId	String	实例ID 示例值：rmq-xxxx
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 创建实例

创建实例

输入示例

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

{
  "InstanceType": "BASIC",
  "Name": "testbuy3",
  "SkuCode": "basic_2k"
}
```

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "InstanceId": "rmq-odrvrqr",
    "RequestId": "32759095-0372-4ec0-ae3a-e5a2759fd0ff"
  }
}
```

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: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub Gitee](#)

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	操作失败。

删除实例

最近更新时间：2024-03-12 01:58:21

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

删除实例

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：DeleteInstance。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
InstanceId	是	String	实例ID 示例值：rocketmq-xxxx

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 删除实例

删除实例

输入示例

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

{
```

```
"InstanceId": "rmq-xxxxx"
}
```

输出示例

```
{
  "Response": {
    "RequestId": "abc"
  }
}
```

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: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub](#) [Gitee](#)

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	操作失败。
ResourceInUse	资源被占用。
ResourceNotFound.Instance	实例不存在。

修改实例属性

最近更新时间：2024-03-12 01:58:20

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

修改实例属性

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：ModifyInstance。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
InstanceId	是	String	实例ID 示例值：rocketmq-xxxx
Name	否	String	实例名称
Remark	否	String	备注信息
SendReceiveRatio	否	Float	消息发送和接收的比例
SkuCode	否	String	调整实例规格的商品代号
MessageRetention	否	Integer	消息保留时长，小时为单位
ScaledTpsEnabled	否	Boolean	是否开启弹性TPS 示例值：false
MaxTopicNum	否	Integer	最大可创建主题数 示例值：100

3. 输出参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 修改实例配置

修改实例配置

输入示例

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

{
  "InstanceId": "rmq-72mo3a9o",
  "Remark": "tests",
  "MessageRetention": 25
}
```

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "RequestId": "becc23c2-b5e1-4df4-9df1-22ba8e5b20c3"
  }
}
```

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: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub Gitee](#)

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	操作失败。
FailedOperation.InstanceNotReady	实例未就绪，请稍后重试。
ResourceUnavailable	资源不可用。
UnsupportedOperation.InstanceTopicNumDowngrade	实例Topic数量不能调整到已使用额度以下。

获取实例列表

最近更新时间：2024-03-12 01:58:20

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

获取实例列表，Filters参数使用说明如下：

1. InstanceName, 名称模糊查询
2. InstanceId, 实例ID查询
3. InstanceType, 实例类型查询，支持多选
4. InstanceStatus, 实例状态查询，支持多选

当使用TagFilters查询时，Filters参数失效。

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：DescribeInstanceList。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
Offset	是	Integer	查询起始位置 示例值：0
Limit	是	Integer	查询结果限制数量 示例值：10
Filters.N	否	Array of Filter	查询条件列表
TagFilters.N	否	Array of TagFilter	标签过滤器

3. 输出参数

参数名称	类型	描述
TotalCount	Integer	查询总数 注意：此字段可能返回 null，表示取不到有效值。 示例值：10
Data	Array of InstanceItem	实例列表

参数名称	类型	描述
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 查询实例列表

查询实例列表

输入示例

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

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

输出示例

```
{
  "Error": null,
  "RequestId": null,
  "Response": {
    "Data": [
      {
        "ExpiryTime": 0,
        "GroupNum": 0,
        "GroupNumLimit": 500,
        "InstanceId": "rmq-4knx4z7p",
        "InstanceName": "tag-test",
        "InstanceStatus": "RUNNING",
        "InstanceType": "EXPERIMENT",
        "MaxMessageDelay": 48,
        "MessageRetention": 72,
        "PayMode": "POSTPAID",
        "Remark": "aabb",
        "ScaledTpsLimit": 0,
        "SkuCode": "experiment_500",
        "TagList": [
          {
            "TagKey": "test-key",
            "TagValue": "tag-value"
          }
        ],
      }
    ]
  }
}
```

```
"TopicNum": 0,
"TopicNumLimit": 50,
"TpsLimit": 500,
"Version": "5.0"
},
{
"ExpiryTime": 0,
"GroupNum": 6,
"GroupNumLimit": 500,
"InstanceId": "rmq-72mo3a9o",
"InstanceName": "breatche-test",
"InstanceStatus": "RUNNING",
"InstanceType": "EXPERIMENT",
"MaxMessageDelay": 48,
"MessageRetention": 24,
"PayMode": "POSTPAID",
"Remark": "cdf",
"ScaledTpsLimit": 0,
"Skucode": "experiment_500",
"TagList": [
{
"TagKey": "tagkey",
"TagValue": "aa"
}
],
"TopicNum": 6,
"TopicNumLimit": 50,
"TpsLimit": 500,
"Version": "5.0"
}
],
"RequestId": "561a34ff-965b-4a2d-852e-973c0b2a6f01",
"TotalCount": 2
}
}
```

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: [GitHub Gitee](#)

- Tencent Cloud SDK 3.0 for Java: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub](#) [Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub](#) [Gitee](#)

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	操作失败。

查询实例信息

最近更新时间：2024-03-12 01:58:21

1. 接口描述

接口请求域名：trocket.tencentcloudapi.com。

查询实例信息

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：DescribeInstance。
Version	是	String	公共参数 ，本接口取值：2023-03-08。
Region	是	String	公共参数 ，详见产品支持的 地域列表 。
InstanceId	是	String	实例ID 示例值：rocketmq-xxxx

3. 输出参数

参数名称	类型	描述
InstanceType	String	实例类型， EXPERIMENT 体验版 BASIC 基础版 PRO 专业版 PLATINUM 铂金版 示例值：BASIC
InstanceId	String	实例ID
InstanceName	String	实例名称
TopicNum	Integer	主题数量
TopicNumLimit	Integer	实例最大主题数量
GroupNum	Integer	消费组数量
GroupNumLimit	Integer	实例最大消费组数量
MessageRetention	Integer	消息保留时间，小时为单位
RetentionUpperLimit	Integer	最大可调整消息保留时间，小时为单位

参数名称	类型	描述
RetentionLowerLimit	Integer	最小可调整消息保留时间, 小时为单位
TpsLimit	Integer	TPS限流值
ScaledTpsLimit	Integer	弹性TPS限流值
MaxMessageDelay	Integer	延迟消息最长时间, 小时为单位
CreatedTime	Integer	创建时间, 秒为单位
SendReceiveRatio	Float	消息发送接收比例
TagList	Array of Tag	标签 注意: 此字段可能返回 null, 表示取不到有效值。
EndpointList	Array of Endpoint	接入点列表 注意: 此字段可能返回 null, 表示取不到有效值。
TopicQueueNumUpperLimit	Integer	主题队列数上限
TopicQueueNumLowerLimit	Integer	主题队列数下限
Remark	String	备注信息
InstanceStatus	String	实例状态
SkuCode	String	实例规格
PayMode	String	计费模式
ScaledTpsEnabled	Boolean	是否开启弹性TPS 示例值: false
RenewFlag	Integer	是否自动续费
ExpiryTime	Integer	到期时间
RoleNumLimit	Integer	角色数量限制
AclEnabled	Boolean	是否开启 ACL 注意: 此字段可能返回 null, 表示取不到有效值。
TopicNumLowerLimit	Integer	topic个数免费额度 注意: 此字段可能返回 null, 表示取不到有效值。 示例值: 0
TopicNumUpperLimit	Integer	最大可设置的topic个数 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, 由服务端生成, 每次请求都会返回 (若请求因其他原因未能抵达服务端, 则该次请求不会获得 RequestId)。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 查询实例

查询实例

输入示例

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

```
{
  "InstanceId": "abc"
}
```

输出示例

```
{
  "Response": {
    "ScaledTpsEnabled": false,
    "InstanceType": "abc",
    "InstanceId": "abc",
    "InstanceName": "abc",
    "TopicNum": 0,
    "TopicNumLimit": 0,
    "GroupNum": 0,
    "GroupNumLimit": 0,
    "MessageRetention": 0,
    "RetentionUpperLimit": 0,
    "RetentionLowerLimit": 0,
    "TpsLimit": 0,
    "ScaledTpsLimit": 0,
    "MaxMessageDelay": 0,
    "CreatedTime": 0,
    "SendReceiveRatio": 0,
    "TagList": [
      {
        "TagKey": "abc",
        "TagValue": "abc"
      }
    ],
    "EndpointList": [
      {
        "Type": "abc",
        "Status": "abc",
        "PayMode": "abc",
        "EndpointUrl": "abc",
        "VpId": "abc",
        "SubnetId": "abc",
        "Bandwidth": 0,
        "IpRules": [
          {
            "Ip": "abc",
            "Allow": true,
```

```
"Remark": "abc"
}
]
}
],
"TopicQueueNumUpperLimit": 0,
"TopicQueueNumLowerLimit": 0,
"Remark": "abc",
"InstanceStatus": "abc",
"SkuCode": "abc",
"PayMode": "abc",
"RequestId": "abc"
}
}
```

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: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Java: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for PHP: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Go: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Node.js: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for .NET: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for C++: [GitHub Gitee](#)
- Tencent Cloud SDK 3.0 for Ruby: [GitHub Gitee](#)

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
ResourceNotFound.Instance	实例不存在。

数据结构

最近更新时间：2024-04-22 01:28:24

ConsumeGroupItem

消费组信息

被如下接口引用：DescribeConsumerGroupList。

名称	类型	必选	描述
InstanceId	String	否	实例ID
ConsumerGroup	String	否	消费组名称
ConsumeEnable	Boolean	否	是否开启消费
ConsumeMessageOrderly	Boolean	否	顺序投递：true 并发投递：false
MaxRetryTimes	Integer	否	最大重试次数
Remark	String	否	备注

CustomMapEntry

map结构返回

被如下接口引用：DescribeMQTTMessage。

名称	类型	必选	描述
Key	String	否	key 注意：此字段可能返回 null，表示取不到有效值。 示例值：key1
Value	String	否	value 注意：此字段可能返回 null，表示取不到有效值。 示例值：value1

Endpoint

接入点信息

被如下接口引用：DescribeInstance。

名称	类型	描述
Type	String	接入点类型，枚举值如下 VPC: VPC; PUBLIC: 公网; INTERNAL: 支撑网; 示例值：VPC
Status	String	状态， OPEN 开启， CLOSE 关闭， PROCESSING 操作中，

名称	类型	描述
PayMode	String	付费类型，仅公网 PREPAID 包年包月 POSTPAID 按量付费 注意：此字段可能返回 null，表示取不到有效值。
EndpointUrl	String	接入点地址 注意：此字段可能返回 null，表示取不到有效值。
VpcId	String	VPC ID 注意：此字段可能返回 null，表示取不到有效值。
SubnetId	String	子网ID 注意：此字段可能返回 null，表示取不到有效值。
Bandwidth	Integer	公网带宽，Mbps为单位 注意：此字段可能返回 null，表示取不到有效值。
IpRules	Array of IpRule	公网放通规则 注意：此字段可能返回 null，表示取不到有效值。

Filter

查询过滤器

被如下接口引用：DescribeConsumerGroupList, DescribeInstanceList, DescribeMQTTInstanceList, DescribeMQTTTopicList, DescribeMQTTUserList, DescribeRoleList, DescribeTopic, DescribeTopicList。

名称	类型	必选	描述
Name	String	是	过滤条件名 示例值：key
Values	Array of String	是	过滤条件的值

InstanceItem

实例列表页中的实例信息

被如下接口引用：DescribeInstanceList。

名称	类型	描述
InstanceId	String	实例ID 示例值：rmq-xxxx
InstanceName	String	实例名称 示例值：test
Version	String	实例版本 示例值：5.0
InstanceType	String	实例类型， EXPERIMENT，体验版 BASIC，基础版 PRO，专业版 PLATINUM，铂金版

名称	类型	描述
InstanceStatus	String	实例状态， RUNNING, 运行中 MAINTAINING, 维护中 ABNORMAL, 异常 OVERDUE, 欠费 DESTROYED, 已删除 CREATING, 创建中 MODIFYING, 变配中 CREATE_FAILURE, 创建失败 MODIFY_FAILURE, 变配失败 DELETING, 删除中
TopicNumLimit	Integer	实例主题数上限 示例值：100
GroupNumLimit	Integer	实例消费组数量上限 示例值：100
PayMode	String	计费模式， POSTPAID, 按量计费 PREPAID, 包年包月 示例值：0
ExpiryTime	Integer	到期时间，秒为单位 注意：此字段可能返回 null，表示取不到有效值。 示例值：100000
Remark	String	备注信息 注意：此字段可能返回 null，表示取不到有效值。 示例值：test
TopicNum	Integer	主题数量
GroupNum	Integer	消费组数量
TagList	Array of Tag	标签列表 注意：此字段可能返回 null，表示取不到有效值。
SkuCode	String	商品规格 示例值：basic_1k
TpsLimit	Integer	TPS限流值 注意：此字段可能返回 null，表示取不到有效值。
ScaledTpsLimit	Integer	弹性TPS限流值 注意：此字段可能返回 null，表示取不到有效值。
MessageRetention	Integer	消息保留时间，小时为单位 注意：此字段可能返回 null，表示取不到有效值。
MaxMessageDelay	Integer	延迟消息最大时长，小时为单位 注意：此字段可能返回 null，表示取不到有效值。

IpRule

IP规则

被如下接口引用：CreateInstance, CreateMQTTInstance, DescribeInstance。

名称	类型	必选	描述
Ip	String	是	IP地址 示例值：0.0.0.0
Allow	Boolean	是	是否允许放行 示例值：true
Remark	String	是	备注信息 注意：此字段可能返回 null，表示取不到有效值。 示例值：remark1

MQTTClientSubscription

MQTT 订阅关系

被如下接口引用：DescribeMQTTClient。

名称	类型	描述
TopicFilter	String	topic 订阅 示例值：mytopic/a/+
Qos	Integer	服务质量等级 示例值：1

MQTTEndpointItem

MQTTEndpoint

被如下接口引用：DescribeMQTTInsPublicEndpoints, DescribeMQTTInsVPCEndpoints。

名称	类型	必选	描述
Type	String	否	类型 注意：此字段可能返回 null，表示取不到有效值。 示例值：mqtt-tcp
Url	String	否	接入点 注意：此字段可能返回 null，表示取不到有效值。 示例值：mqtt-12345-cd.mqtt.public.tencenttdmq.com:1883
VpcId	String	否	vpc信息 注意：此字段可能返回 null，表示取不到有效值。 示例值：vpc-23dada
SubnetId	String	否	子网信息 注意：此字段可能返回 null，表示取不到有效值。 示例值：subnet-dadaf
Host	String	否	主机 注意：此字段可能返回 null，表示取不到有效值。 示例值：mqtt-12345-cd.mqtt.public.tencenttdmq.com
Port	Integer	否	端口 注意：此字段可能返回 null，表示取不到有效值。 示例值：1883

名称	类型	必选	描述
Ip	String	否	接入点ip 注意：此字段可能返回 null，表示取不到有效值。 示例值：127.0.0.1

MQTTInstanceItem

MQTT 实例信息

被如下接口引用：DescribeMQTTInstanceList。

名称	类型	描述
InstanceId	String	实例ID 示例值：rmq-xxxx
InstanceName	String	实例名称 示例值：test
Version	String	实例版本 示例值：5.0
InstanceType	String	实例类型， EXPERIMENT，体验版 BASIC，基础版 PRO，专业版 PLATINUM，铂金版 示例值：EXPERIMENT
InstanceStatus	String	实例状态， RUNNING，运行中 MAINTAINING，维护中 ABNORMAL，异常 OVERDUE，欠费 DESTROYED，已删除 CREATING，创建中 MODIFYING，变配中 CREATE_FAILURE，创建失败 MODIFY_FAILURE，变配失败 DELETING，删除中 示例值：RUNNING
TopicNumLimit	Integer	实例主题数上限 示例值：100
Remark	String	备注信息 注意：此字段可能返回 null，表示取不到有效值。 示例值：test
TopicNum	Integer	主题数量 示例值：2
SkuCode	String	商品规格 示例值：basic_1k
TpsLimit	Integer	弹性TPS限流值 注意：此字段可能返回 null，表示取不到有效值。 示例值：100

名称	类型	描述
CreateTime	Integer	创建时间 注意：此字段可能返回 null，表示取不到有效值。 示例值：1713164969433
SubscriptionNumLimit	Integer	订阅关系上限 注意：此字段可能返回 null，表示取不到有效值。 示例值：1000
ClientNumLimit	Integer	客户端连接数上限 注意：此字段可能返回 null，表示取不到有效值。 示例值：1000

MQTTMessageItem

消息记录

被如下接口引用：DescribeMQTTMessageList。

名称	类型	描述
MsgId	String	消息ID 注意：此字段可能返回 null，表示取不到有效值。 示例值：test
Tags	String	消息tag 注意：此字段可能返回 null，表示取不到有效值。 示例值：tag1
Keys	String	消息key 注意：此字段可能返回 null，表示取不到有效值。 示例值：value1
ProducerAddr	String	客户端地址 注意：此字段可能返回 null，表示取不到有效值。 示例值：127.0.0.1
ProduceTime	String	消息发送时间 注意：此字段可能返回 null，表示取不到有效值。 示例值：1713164969433
DeadLetterResendTimes	Integer	死信重发次数 注意：此字段可能返回 null，表示取不到有效值。 示例值：2
DeadLetterResendSuccessTimes	Integer	死信重发成功次数 注意：此字段可能返回 null，表示取不到有效值。 示例值：2
SubTopic	String	子topic 注意：此字段可能返回 null，表示取不到有效值。 示例值：subtopic
Qos	String	消息质量等级 注意：此字段可能返回 null，表示取不到有效值。 示例值：1

MQTTProductSkulItem

MQTT ProductSkulItem

被如下接口引用：DescribeMQTTProductSKUList。

名称	类型	描述
InstanceType	String	类型 注意：此字段可能返回 null，表示取不到有效值。 示例值：BASIC
SkuCode	String	cide 注意：此字段可能返回 null，表示取不到有效值。 示例值：basic_1000
OnSale	Boolean	sale 注意：此字段可能返回 null，表示取不到有效值。 示例值：1
TopicNumLimit	Integer	topic num限制 注意：此字段可能返回 null，表示取不到有效值。 示例值：100
TpsLimit	Integer	tps 注意：此字段可能返回 null，表示取不到有效值。 示例值：1000
ClientNumLimit	Integer	客户端连接数 注意：此字段可能返回 null，表示取不到有效值。 示例值：1000
SubscriptionNumLimit	Integer	订阅数限制 注意：此字段可能返回 null，表示取不到有效值。 示例值：1000
ProxySpecCore	Integer	代理核 注意：此字段可能返回 null，表示取不到有效值。 示例值：2
ProxySpecMemory	Integer	代理内存 注意：此字段可能返回 null，表示取不到有效值。 示例值：2
ProxySpecCount	Integer	代理总数 注意：此字段可能返回 null，表示取不到有效值。 示例值：2

MQTTTopicItem

MQTT 主题详情

被如下接口引用：DescribeMQTTTopicList。

名称	类型	描述
InstanceId	String	实例 ID 示例值：mqtt-xxx
Topic	String	主题名称 示例值：mytopic

名称	类型	描述
Remark	String	主题描述 注意：此字段可能返回 null，表示取不到有效值。 示例值：remark1

MQTTUserItem

MQTT集群用户信息

被如下接口引用：DescribeMQTTUserList。

名称	类型	描述
Username	String	用户名 示例值：user1
Password	String	密码 示例值：pd
PermRead	Boolean	是否开启消费 示例值：true
PermWrite	Boolean	是否开启生产 示例值：true
Remark	String	备注信息 示例值：remark1
CreatedTime	Integer	创建时间，秒为单位 示例值：1712739629775
ModifiedTime	Integer	修改时间，秒为单位 示例值：1712739629775

PacketStatistics

MQTT客户端监控

被如下接口引用：DescribeMQTTClient。

名称	类型	描述
MessageType	String	类型 注意：此字段可能返回 null，表示取不到有效值。 示例值：CONNECT
Qos	Integer	服务质量 注意：此字段可能返回 null，表示取不到有效值。 示例值：0
Count	Integer	指标值 注意：此字段可能返回 null，表示取不到有效值。 示例值：0

PublicAccessRule

公网访问安全规则

被如下接口引用：CreateMQTTInsPublicEndpoint, DescribeMQTTInsPublicEndpoints, ModifyMQTTInsPublicEndpoint。

名称	类型	必选	描述
IpRule	String	是	ip网段信息 注意：此字段可能返回 null，表示取不到有效值。 示例值：0.0.0.0/0
Allow	Boolean	是	允许或者拒绝 注意：此字段可能返回 null，表示取不到有效值。 示例值：true
Remark	String	否	备注信息 注意：此字段可能返回 null，表示取不到有效值。 示例值：test

RoleItem

角色信息

被如下接口引用：DescribeRoleList。

名称	类型	描述
RoleName	String	角色名称 示例值：test
AccessKey	String	Access Key
SecretKey	String	Secret Key
PermRead	Boolean	是否开启消费
PermWrite	Boolean	是否开启生产
Remark	String	备注信息
CreatedTime	Integer	创建时间，秒为单位
ModifiedTime	Integer	修改时间，秒为单位

SourceClusterGroupConfig

消费组配置信息

被如下接口引用：ImportSourceClusterConsumerGroups。

名称	类型	必选	描述
GroupName	String	是	消费组名称 注意：此字段可能返回 null，表示取不到有效值。
Remark	String	否	备注信息 注意：此字段可能返回 null，表示取不到有效值。
Imported	Boolean	否	是否已导入，作为入参时无效 注意：此字段可能返回 null，表示取不到有效值。
Namespace	String	否	命名空间，仅4.x集群有效 注意：此字段可能返回 null，表示取不到有效值。

名称	类型	必选	描述
ImportStatus	String	否	导入状态 Unknown 未知 Success 成功 Failure 失败 AlreadyExists 已存在 注意：此字段可能返回 null，表示取不到有效值。

SourceClusterTopicConfig

源集群主题配置

被如下接口引用：ImportSourceClusterTopics。

名称	类型	必选	描述
TopicName	String	是	主题名称 注意：此字段可能返回 null，表示取不到有效值。
TopicType	String	是	主题类型， 5.x版本 UNSPECIFIED 未指定 NORMAL 普通消息 FIFO 顺序消息 DELAY 延迟消息 TRANSACTION 事务消息 4.x版本 Normal 普通消息 PartitionedOrder 分区顺序消息 Transaction 事务消息 DelayScheduled 延时消息 注意：此字段可能返回 null，表示取不到有效值。
QueueNum	Integer	是	队列数 注意：此字段可能返回 null，表示取不到有效值。
Remark	String	否	备注信息 注意：此字段可能返回 null，表示取不到有效值。
Imported	Boolean	否	是否已导入，作为入参时无效 注意：此字段可能返回 null，表示取不到有效值。
Namespace	String	否	命名空间，仅4.x集群有效 注意：此字段可能返回 null，表示取不到有效值。
ImportStatus	String	否	导入状态， Unknown 未知， AlreadyExists 已存在， Success 成功， Failure 失败 注意：此字段可能返回 null，表示取不到有效值。

StatisticsReport

MQTT客户端数据流量统计

被如下接口引用：DescribeMQTTClient。

名称	类型	描述
Bytes	Integer	字节数 注意：此字段可能返回 null，表示取不到有效值。 示例值：10
Items	Array of PacketStatistics	监控指标 注意：此字段可能返回 null，表示取不到有效值。

SubscriptionData

主题与消费组的订阅关系数据

被如下接口引用：DescribeTopic。

名称	类型	描述
InstanceId	String	实例ID 注意：此字段可能返回 null，表示取不到有效值。 示例值：rmq-xxx
Topic	String	主题名称 注意：此字段可能返回 null，表示取不到有效值。 示例值：test
TopicType	String	主题类型 注意：此字段可能返回 null，表示取不到有效值。
TopicQueueNum	Integer	单个节点上主题队列数 注意：此字段可能返回 null，表示取不到有效值。
ConsumerGroup	String	消费组名称 注意：此字段可能返回 null，表示取不到有效值。
IsOnline	Boolean	是否在线 注意：此字段可能返回 null，表示取不到有效值。
ConsumeType	String	消费类型 注意：此字段可能返回 null，表示取不到有效值。
SubString	String	订阅规则 注意：此字段可能返回 null，表示取不到有效值。
ExpressionType	String	过滤类型 注意：此字段可能返回 null，表示取不到有效值。
Consistency	Integer	订阅一致性 注意：此字段可能返回 null，表示取不到有效值。
ConsumerLag	Integer	消费堆积 注意：此字段可能返回 null，表示取不到有效值。
LastUpdateTime	Integer	最后消费进度更新时间，秒为单位 注意：此字段可能返回 null，表示取不到有效值。
MaxRetryTimes	Integer	最大重试次数 注意：此字段可能返回 null，表示取不到有效值。
ConsumeMessageOrderly	Boolean	是否顺序消费 注意：此字段可能返回 null，表示取不到有效值。

名称	类型	描述
MessageModel	String	消费模式： BROADCASTING 广播模式； CLUSTERING 集群模式； 注意：此字段可能返回 null，表示取不到有效值。 示例值：CLUSTERING

Tag

标签数据

被如下接口引用：CreateInstance, CreateMQTTInstance, DescribeInstance, DescribeInstanceList。

名称	类型	必选	描述
TagKey	String	是	标签名称 注意：此字段可能返回 null，表示取不到有效值。 示例值：123
TagValue	String	是	标签值 注意：此字段可能返回 null，表示取不到有效值。 示例值：abc

TagFilter

标签过滤器

被如下接口引用：DescribeInstanceList。

名称	类型	必选	描述
TagKey	String	是	标签键名称 示例值：test
TagValues	Array of String	是	标签值列表

TopicItem

列表上的主题信息

被如下接口引用：DescribeTopicList。

名称	类型	必选	描述
InstanceId	String	否	实例ID 示例值：rmq-xxxx
Topic	String	否	主题名称 示例值：test
TopicType	String	否	主题类型
QueueNum	Integer	否	队列数量 示例值：3
Remark	String	否	描述 注意：此字段可能返回 null，表示取不到有效值。

名称	类型	必选	描述
ClusterIdV4	String	否	4.x的集群id 注意：此字段可能返回 null，表示取不到有效值。
NamespaceV4	String	否	4.x的命名空间 注意：此字段可能返回 null，表示取不到有效值。
TopicV4	String	否	4.x的主题名称 注意：此字段可能返回 null，表示取不到有效值。
FullNamespaceV4	String	否	4.x的完整命名空间 注意：此字段可能返回 null，表示取不到有效值。
MsgTTL	Integer	否	消息保留时长 注意：此字段可能返回 null，表示取不到有效值。 示例值：72

VpcInfo

VPC信息

被如下接口引用：CreateInstance, CreateMQTTInstance。

名称	类型	必选	描述
VpcId	String	是	VPC ID 示例值：vpc-xxxx
SubnetId	String	是	子网ID 示例值：subnet-xxxx

错误码

最近更新时间：2024-03-07 01:26:43

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
FailedOperation.InstanceNotReady	实例未就绪，请稍后重试。
LimitExceeded.TopicNum	已达到主题数上限。
OperationDenied	操作被拒绝。
ResourceNotFound.Client	客户端不存在。
ResourceNotFound.Group	消费组不存在，请检查后重试。
ResourceNotFound.Instance	实例不存在。
ResourceNotFound.Message	消息不存在。

错误码	说明
ResourceNotFound.Role	角色不存在，请检查后重试。
ResourceNotFound.Topic	主题不存在，请检查后重试。
UnsupportedOperation.InstanceTopicNumDowngrade	实例Topic数量不能调整到已使用额度以下。
UnsupportedOperation.ResourceAlreadyExists	资源已存在，请检查后重试。