智能保险助手

API 文档

腾讯云

文档目录

API 文档
更新历史

最近更新时间：2022-06-27 06:06:05

第 14 次发布

发布时间：2022-06-27 06:06:02

本次发布包含了以下内容：

改善已有的文档。

修改数据结构：

- OcrRecognise
  - 新增成员：Field

第 13 次发布

发布时间：2022-06-16 06:03:27

本次发布包含了以下内容：

改善已有的文档。

新增数据结构：

- Location
- OcrRecognise
- Point

修改数据结构：

- StructureResultObject
  - 新增成员：ResultFields

第 12 次发布

发布时间：2022-02-18 16:00:08

本次发布包含了以下内容：

改善已有的文档。

修改数据结构：

- UnderwriteItem
  - 新增成员：Range, ReportDate, FileType, InspectProject, Unit, OriginName, YinYang

第 11 次发布

发布时间：2022-02-18 14:29:36

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- DescribeQualityScore

第 10 次发布

发布时间：2022-02-17 08:07:32

本次发布包含了以下内容：

改善已有的文档。

修改数据结构：

- StructureResultObject
  - 新增成员：TaskFiles

第 9 次发布

发布时间：2022-01-20 08:06:41

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- AddSubStructureTasks
- DescribeReportClassify
- DescribeUnderwriteTask

新增数据结构：

- ClassifiedReports
- UnderwriteConclusion
- UnderwriteOutput

第 8 次发布

发布时间：2021-12-13 08:06:17

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- CreateAutoClassifyStructureTask

新增数据结构：

- CreateAutoClassifyStructureTaskInfo

第 7 次发布

发布时间：2021-11-23 08:04:46

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- UploadMedicalFile

第 6 次发布

发布时间：2021-09-30 08:02:25

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- CreateUnderwriteTaskById
- DescribeMachineUnderwrite
- DescribeStructureDifference

删除接口：

- CreateStructureTaskTest
- DescribeStructureTaskResultTest

新增数据结构：

- InsuranceResult
- MachinePredict
- MachineUnderwriteOutput
- PerStructDifference
- StructureModifyItem
- StructureOneItem
- UnderwriteItem

第 5 次发布

发布时间：2021-09-24 09:54:37

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- DescribeStructureResult
  - 新增出参：MainTaskId

修改数据结构：

- StructureResultObject
  - 新增成员：SubTaskId

第 4 次发布

发布时间：2021-09-23 08:05:48

本次发布包含了以下内容：

改善已有的文档。

新增数据结构：

- ClassifyInfo

修改数据结构：

- ResultObject
  - 新增成员：ReportType

第 3 次发布

发布时间：2021-08-03 08:12:34

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- CreateStructureTaskTest
- DescribeStructureTaskResultTest

第 2 次发布

发布时间：2021-07-01 08:11:12

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- DescribeStructureResult

修改接口：

- CreateStructureTask
  - 新增入参：CallbackUrl

新增数据结构：

- StructureResultObject

第 1 次发布

发布时间：2021-05-08 15:29:47

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- CreateStructureTask
- DescribeStructCompareData
- DescribeStructureTaskResult

新增数据结构：

- CompareMetricsData
- CreateStructureTaskInfo
- ResultObject
- ReviewDataTaskInfo

简介

最近更新时间：2021-06-23 11:40:09

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

智能保险助手可以将各类医疗票据结构化，辅助核保师给出AI建议的核保结论，帮助保险企业长期积累用户数据的同时提高核保效率。

API 概览

最近更新时间：2023-05-17 01:16:02

文件上传相关接口

接口名称	接口功能	频率限制（次/秒）
UploadMedicalFile	上传医疗影像文件	20

结构化任务相关接口

接口名称	接口功能	频率限制（次/秒）
CreateStructureTask	新建结构化任务	20
AddSubStructureTasks	结构化增量子任务	20
DescribeStructureResult	查询结构化结果接口	20
DescribeReportClassify	报告自动分类	20
CreateAutoClassifyStructureTask	新建自动分类结构化任务	20
DescribeQualityScore	获取图片质量分	20
DescribeStructureTaskResult	获取结构化结果接口	20

结构化对比相关接口

接口名称	接口功能	频率限制（次/秒）
DescribeStructCompareData	结构化对比查询	20
DescribeStructureDifference	结构化复核差异查询	20

核保相关接口

接口名称	接口功能	频率限制（次/秒）
CreateUnderwriteTaskById	根据结构化任务ID创建核保任务	20
DescribeMachineUnderwrite	查询机器核保任务数据	20
DescribeUnderwriteTask	查询核保任务数据	20

调用方式
公共参数

最近更新时间：2023-12-28 01:09:57

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入参公共参数 Version 的说明。例如云服务器的版本 2017-03-12。
Authorization	String	是	HTTP 标准身份认证头部字段，例如： TC3-HMAC-SHA256 Credential=AKIDEXAMPLE/Date/service/tc3_request, SignedHeaders=content-type;host, Signature=fe5f80f77d5fa3beca038a248ff027d0445342fe2855ddc963176630326f1024 其中， - TC3-HMAC-SHA256：签名方法，目前固定取该值； - Credential：签名凭证，AKIDEXAMPLE 是 SecretId；Date 是 UTC 标准时间的日期，取值需要和公共参数 X-TC-Timestamp 换算的 UTC 标准时间日期一致；service 为具体产品名，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 cii；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（有时会称作 HmacSHA256 和 HmacSHA1），公共参数需要统一放到请求串中，完整介绍详见文档

参数名称	类型	必选	描述
Action	String	是	操作的接口名称。取值参考接口文档中输入参数章节关于公共参数 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入参公共参数 Version 的说明。例如云服务器的版本 2017-03-12。
SignatureMethod	String	否	签名方式，目前支持 HmacSHA256 和 HmacSHA1。只有指定此参数为 HmacSHA256 时，才使用 HmacSHA256 算法验证签名，其他情况均使用 HmacSHA1 验证签名。
Token	String	否	即 安全凭证服务 所颁发的临时安全凭证中的 Token，使用时需要将 SecretId 和 SecretKey 的值替换为临时安全凭证中的 TmpSecretId 和 TmpSecretKey。使用长期密钥时不能设置此 Token 字段。
Language	String	否	指定接口返回的语言，仅部分接口支持此参数。取值：zh-CN，en-US。zh-CN 返回中文，en-US 返回英文。

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

HTTP GET 请求结构示例：

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

Host: cvm.tencentcloudapi.com
```

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
```

签名方法 v3

最近更新时间：2023-12-27 01:11:05

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

根据以上规则，示例中得到的规范请求串如下：

```
POST
/

content-type:application/json; charset=utf-8
host:cvm.tencentcloudapi.com
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 是二进制的数据，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，也可能会提供打印友好的十六进制格式的返回值，此处使用的是二进制格式。

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

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.utcfromtimestamp(timestamp).strftime("%Y-%m-%d")
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}\nx-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": ["\u672a\u547d\u540d"], "Name": "instance-name"}]}'
hashed_request_payload = Digest::SHA256.hexdigest(payload)
canonical_request = [
http_request_method,
canonical_uri,
canonical_querystring,
canonical_headers,
signed_headers,
hashed_request_payload,
].join("\n")

puts canonical_request

# ************* 步骤 2：拼接待签名字符串 *************
credential_scope = date + '/' + service + '/' + 'tc3_request'
hashed_request_payload = Digest::SHA256.hexdigest(canonical_request)
string_to_sign = [
algorithm,
timestamp.to_s,
credential_scope,
hashed_request_payload,
].join("\n")
puts string_to_sign

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

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

```
puts authorization

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

DotNet

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

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

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

public static Dictionary<String, String> BuildHeaders(string secretid,
string secretkey, string service, string endpoint, string region,
string action, string version, DateTime date, string requestPayload)
{
string datestr = date.ToString("yyyy-MM-dd");
DateTime startTime = new DateTime(1970, 1, 1, 0, 0, 0, 0, DateTimeKind.Utc);
```

```
long requestTimestamp = (long)Math.Round((date - startTime).TotalMilliseconds, MidpointRounding.AwayFromZero) / 1000;
// ************* 步骤 1：拼接规范请求串 *************
string algorithm = "TC3-HMAC-SHA256";
string httpRequestMethod = "POST";
string canonicalUri = "/";
string canonicalQueryString = "";
string contentType = "application/json";
string canonicalHeaders = "content-type:" + contentType + "; charset=utf-8\n"
+ "host:" + endpoint + "\n"
+ "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 = algorithm + " "
+ "Credential=" + secretid + "/" + credentialScope + ", "
+ "SignedHeaders=" + signedHeaders + ", "
+ "Signature=" + signature;
Console.WriteLine(authorization);

Dictionary<string, string> headers = new Dictionary<string, string>();
headers.Add("Authorization", authorization);
headers.Add("Host", endpoint);
headers.Add("Content-Type", contentType + "; charset=utf-8");
headers.Add("X-TC-Timestamp" , requestTimestamp.ToString());
```

```
headers.Add("X-TC-Version", version);
headers.Add("X-TC-Action", action);
headers.Add("X-TC-Region", region);
return headers;
}
public static void Main(string[] args)
{
// 密钥参数
// 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, 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++)
{
snprintf(buf, sizeof(buf), "%02x", hash[i]);
NewString = NewString + buf;
}
return NewString;
}

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

HMAC_CTX *h;
#if OPENSSL_VERSION_NUMBER < 0x10100000L
HMAC_CTX hmac;
HMAC_CTX_init(&hmac);
h = &hmac;
#else
h = HMAC_CTX_new();
#endif
```

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

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

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

return (ss.str());
}

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

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

int main()
{
// 密钥参数
// 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++)
{
snprintf(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);

#if 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）。

智能保险助手页面集成说明

最近更新时间：2022-02-16 08:03:26

智能保险助手页面集成说明

1. 接入信息录入

进入【 https://cii.tencent.cn/ 】通过腾讯云账号登陆，进入【 页面集成申请 】登记集成信息。Token地址对应OauthToken接口，UserInfo地址对应OauthUser接口。如果涉及到结构化/复核/核保异步通知，复核回调地址也需要登记。如果还不确定，登记可以不填，对接时候再更新。

2. OA集成，接入方需提供鉴权接口

核保结构化等页面集成需要账号打通。集成方需要提供OAuth相关接口，saas后端oauth验证成功后方可使用。以下接口仅供参考，具体接口需集成方提供。

2.1 OauthToken

授权码获取accesstoken的接口。授权码由集成方生成，建议结合

输入

字段名称	type	desc
Code	string	授权码。由集成方生成，建议生成策略结合openid生成，有有效期
GrantType	string	固定为 authorization_code
ClientId	string	集成方提供
ClientSecret	string	集成方提供

输出

字段名称	type	desc
AccessToken	string	登陆token
Expire	string	过期时间
RefreshToken	string	用于刷新 Access Token 的 Refresh Token

2.2 OauthUser

accesstoken获取用户信息

输入

字段名称	type	desc

字段名称	type	desc
AccessToken	string	登陆token

输出

字段名称	type	desc
UserName	string	用户名
UserId	int	用户唯一表示

3. 前端集成

3.1 对接环境/访问域名

访问域名（host）:https://cii.tencent.cn/

3.2 iframe页面集成

页面名称	参数说明	接入地址（iframe src）
结构化复核	structureMainTaskId: 结构化主任号code: GenerateCode生成openId: 开通企业服务生成	https://{host}/tasks/{structureMainTaskId}/review?code={code}&openId={openId}
结构化对比	structureMainTaskId:结构化主任号code: GenerateCode生成openId:开通企业服务生成	https://{host}/tasks/{structureMainTaskId}/compare?code={code}&openId={openId}
核保复核	underwriteTaskId: 核保任号code: GenerateCode生成openId: 开通企业服务生成	https://{host}/underwrites/{underwriteTaskId}/review?operate=check&code={code}&openId={openId}

4. 回调接口说明

4.1 结构化复核状态回调接口

接口地址

客户侧提供（HTTP+POST+JSON的方式）

请求参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，每次请求都会返回。定位问题时需要提供该次请求的 RequestId。
MainTaskId	String	主任务ID，不重复且唯一
SubTaskId	String	子任务ID，不重复且唯一，复核保存时提供
Status	String	复核状态枚举Done/Doing/Saved

请求示例

```
{
"Response":{
"RequestId":"22dfcc05-1ba1-49b4-a751-f5611cdb3420",
"MainTaskId":"Mshbylkq8buo",
"SubTaskId":"Sr5ifsludlhd",
"Status":"Done/Doing/Saved"
}
}
```


4.2 结构化结果回调接口

接口地址

客户侧提供（HTTP+POST+JSON的方式），创建任务时通过CallbackUrl参数传递

请求参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，每次请求都会返回。定位问题时需要提供该次请求的 RequestId。
MainTaskId	String	主任务ID，不重复且唯一
Status	Integer	结果状态：0：返回成功2：结果生成失败
Results	Set of ResultObject	结构化识别结果数组，每个数组元素对应一个图片的结构化结果，顺序和输入参数的ImageList或FileList对应。

ResultObject

参数名称	类型	描述
Code	Integer	状态码：0代表成功；1代表结果为空；2代表下载结果失败
StructureResult	String	由结构化算法结构化json转换的字符串，具体协议参见算法结构化结果协议
TaskType	String	报告类型枚举
SubTaskId	String	子任务ID
TaskFiles	Array of String	任务文件数组

请求示例


```
{
"Response":{
"RequestId":"22dfcc05-1ba1-49b4-a751-f5611cdb3420",
"MainTaskId":"asd23-25kd-dl234",
"Status":0,
"Results":[
{
"SubTaskId":"ssss",
"Code":0,
"StructureResult":"{\"abc\":\"123\"}",
"TaskType":"HealthReport",
```

```
"TaskFiles": [
"700000198392/original_upload_dir/700000198392_cba00b8e-4124-4ab3-8ae0-ab9d00583b3d.jpg"
]
},
{
"SubTaskId":"ssss2",
"Code":0,
"StructureResult":"{\"abcd\":\"123\"}",
"TaskType":"HealthReport",
"TaskFiles": [
"700000198392/original_upload_dir/700000198392_cba00b8e-4124-4ab3-8ae0-ab9d00583b3d.jpg"
]
}
]
}
}
```


4.3 核保结果回调

客户侧提供（HTTP+POST+JSON的方式），创建任务时通过CallbackUrl参数传递

请求参数

参数名称	类型	描述
RequestId	String	唯一请求 ID，每次请求都会返回。定位问题时需要提供该次请求的 RequestId。
Uin	String	主账号ID
SubAccountUin	String	操作人子账户ID
PolicyId	String	保单ID
MainTaskId	String	任务ID
UnderwriteTaskId	String	核保任务ID
Status	Integer	结果状态：0：返回成功1：结果未生成2：结果生成失败
ReviewType	String	MANUAL：手动复核MACHINE：机器结果
UnderwriteResults	Array of UnderwriteResult	核保结果

UnderwriteResult

核保结果，包含多个用户的结果。当前只支持一个用户

名称	类型	描述
CustomerId	String	客户号
CustomerName	String	客户名
MachineResult	Array of Object	机器结论

输出示例

```
{
"Response": {
"RequestId": "asdfadasdfasdf",
"Uin": "123xxxxx",
"SubAccountUin": "123xxxxx",
"PolicyId": "123",
"MainTaskId": "Task-1",
"UnderwriteTaskId": "u-task-1",
"UnderwriteResults":[
{
"CustomerId": "user-1",
"CustomerName": "用户A",
"MachineResult": []
},
{
"CustomerId": "user-1",
"CustomerName": "用户A",
"MachineResult": [] // 详细示例见下方
}
]
}
}
```

MachineResult字段示例

```
[
{
"InsuranceType":"重疾险",
"Result":[
{
"Title":"AI决策",
"Conclusion":"承保",
"Explanation":[
{
"Name":"乳腺增生",
"Result":"承保",
"Value":""
}
],
"Disease":[],
"Laboratory":[]
},
{
"Title":"核保结论1",
"Conclusion":"承保",
"Explanation":"",
"Disease":[
{
"Name":"疾病风险指数",
```

```
"Result":"标体",
"Value":""
}
],
"Laboratory":[
{
"Name":"BMI",
"Result":"",
"Value":""
},
{
"Name":"低密度脂蛋白",
"Result":"低胆固醇血症",
"Value":"1.75"
},
{
"Name":"再保2化验指标风险指数",
"Result":"104",
"Value":"0"
},
{
"Name":"总胆固醇",
"Result":"-10",
"Value":"4.09"
}
]
},
{
"Title":"核保结论2",
"Conclusion":"承保",
"Explanation":[],
"Disease":[],
"Laboratory":[]
}
]
},
{
"InsuranceType":"寿险",
"Result":[
{
"Title":"AI决策",
"Conclusion":"",
"Explanation":[],
"Disease":[],
"Laboratory":[]
},
{
"Title":"核保结论1",
"Conclusion":"承保",
"Explanation":[],
```

```
"Disease":[],
"Laboratory":[]
},
{
"Title":"核保结论2",
"Conclusion":"承保",
"Explanation":[],
"Disease":[],
"Laboratory":[]
}
]
},
{
"InsuranceType":"意外险",
"Result":[
{
"Title":"AI决策",
"Conclusion":"",
"Explanation":[],
"Disease":[],
"Laboratory":[]
},
{
"Title":"核保结论1",
"Conclusion":"承保",
"Explanation":[],
"Disease":[],
"Laboratory":[]
},
{
"Title":"核保结论2",
"Conclusion":"承保",
"Explanation":[],
"Disease":[],
"Laboratory":[]
}
]
}
]
```


5. 页面在IE浏览器中集成方案说明

文档模式：document.documentMode

体现为 html meta tag: <meta http-equiv="X-UA-Compatible" content="IE=edge/11/10/9/8/7/6/5">

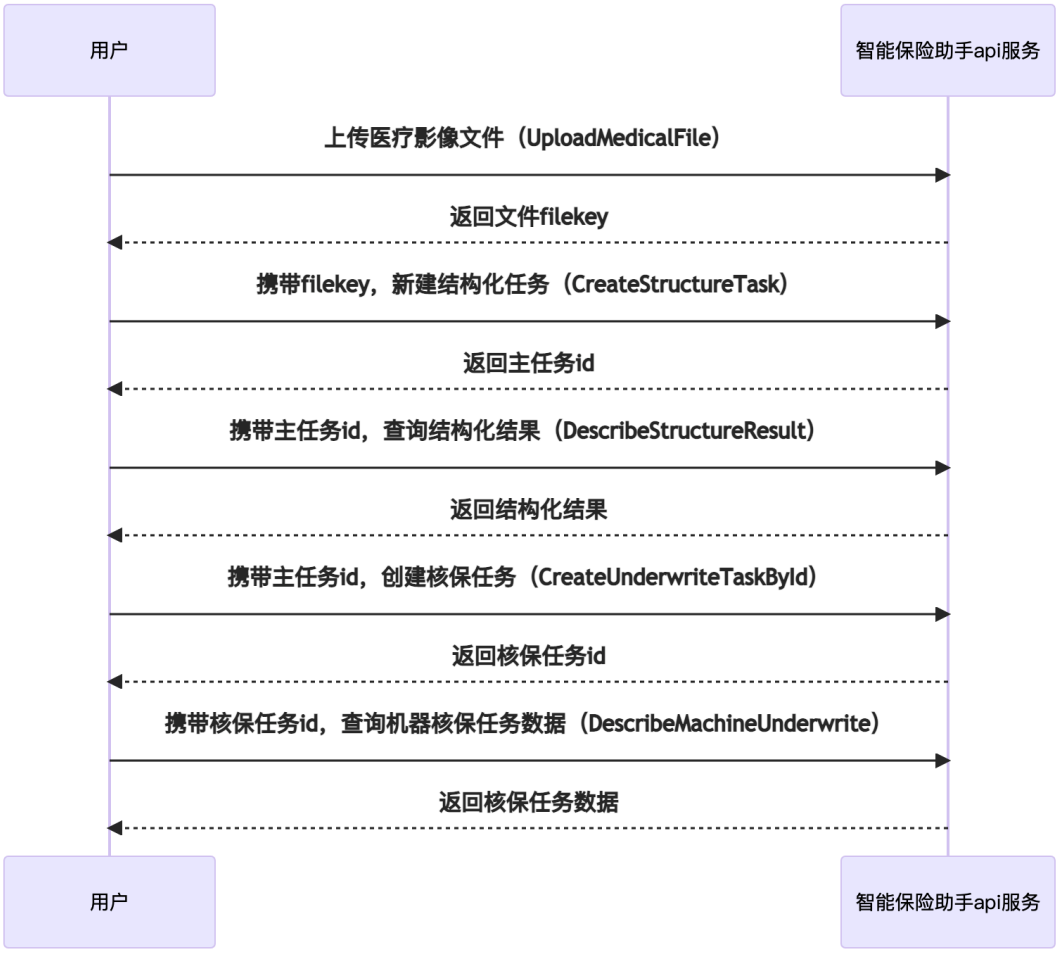
集成方式	客户页面文档模式 < 11	客户页面文档模式 >= 11
iframe嵌入到客户当前页	不支持	支持
iframe嵌入到新标签页	可通过修改文档模式支持	支持

集成方式	客户页面文档模式 < 11	客户页面文档模式 >= 11
url打开新标签（暂未开放）	不存在客户文档模式，由cii控制文档模式来支持	不存在客户文档模式，由cii控制文档模式来支持

API使用指引

最近更新时间：2022-02-18 14:29:40

API使用流程



文件上传相关接口
获取图片质量分

最近更新时间：2023-11-30 01:53:23

1. 接口描述

接口请求域名： cii.tencentcloudapi.com 。

获取图片质量分

默认接口请求频率限制：20次/秒。

注意：本接口只能使用 POST multipart/form-data 调用。

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：DescribeQualityScore。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	否	String	公共参数，本接口不需要传递此参数。
File	是	Binary	文件二进制数据

3. 输出参数

参数名称	类型	描述
QualityScore	Float	质量分 示例值：1
RequestId	String	唯一请求 ID，每次请求都会返回。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 获取图片质量分

输入示例

```
POST / HTTP/1.1
Host: cii.tencentcloudapi.com
Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryLeutxxTRflJusIPV
X-TC-Action: DescribeQualityScore
<公共请求参数>
```

```
------WebKitFormBoundaryLeutxxTRflJusIPV
Content-Disposition: form-data; name="File"; filename="检验报告.jpg"
Content-Type: image/jpg


------WebKitFormBoundaryLeutxxTRflJusIPV--
```

输出示例

```
{
"Response": {
"RequestId": "9317f6e4-6636-41a0-bf24-89ad9e4877f2",
"QualityScore": 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 Node.js: 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
AuthFailure	CAM签名/鉴权错误。
FailedOperation	操作失败。
InternalError	内部错误。

错误码	描述
InvalidParameter	参数错误。
InvalidParameterValue	参数取值错误。
LimitExceeded	超过配额限制。
MissingParameter	缺少参数错误。
OperationDenied	操作被拒绝。
RequestLimitExceeded	请求的次数超过了频率限制。

上传医疗影像文件

最近更新时间：2023-11-30 01:53:26

1. 接口描述

接口请求域名： cii.tencentcloudapi.com 。

上传医疗影像文件，可以用来做结构化。

默认接口请求频率限制：20次/秒。

注意：本接口只能使用 POST multipart/form-data 调用。

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：UploadMedicalFile。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	否	String	公共参数，本接口不需要传递此参数。
File	否	Binary	文件的字节内容。File与FileURL有一个不为空即可，若FileURL参数也存在，会只取File的内容。
FileURL	否	String	文件的URL地址。File与FileURL不能同时为空，若File参数也存在，会只取File的内容。 示例值：https://dss0.bdstatic.com/5aV1bjqh_Q23odCf/static/superman/img/s

3. 输出参数

参数名称	类型	描述
FileKey	String	文件存储的key，可以用来创建结构化任务。 注意：此字段可能返回 null，表示取不到有效值。 示例值：6540556601/original_upload_dir/6540556601_syon08sl1pojv099a0.png
RequestId	String	唯一请求 ID，每次请求都会返回。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 上传医疗影像文件接口示例

输入示例

```
POST / HTTP/1.1
Host: cii.tencentcloudapi.com
Content-Type: multipart/form-data; boundary=----WebKitFormBoundaryLeutxxTRflJusIPV
```

```
X-TC-Action: UploadMedicalFile
<公共请求参数>

------WebKitFormBoundaryLeutxxTRflJusIPV
Content-Disposition: form-data; name="File"; filename="体检报告.pdf"
Content-Type: application/pdf



------WebKitFormBoundaryLeutxxTRflJusIPV--
```

输出示例

```
{
"Response": {
"RequestId": "9317f6e4-6636-41a0-bf24-89ad9e4877f2",
"FileKey": "6540556601/original_upload_dir/6540556601_9317f6e4-6636-41a0-bf24-89ad9e4877f2.png"
}
}
```

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 Node.js: 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
AuthFailure	CAM签名/鉴权错误。
AuthFailure.Enterprise	企业认证失败。

错误码	描述
AuthFailure.Personal	个人认证失败。
FailedOperation	操作失败。
InternalError	内部错误。
InvalidParameter	参数错误。
InvalidParameterValue	参数取值错误。

结构化任务相关接口
新建结构化任务

最近更新时间：2023-11-30 01:53:23

1. 接口描述

接口请求域名： cii.tencentcloudapi.com 。

本接口(CreateStructureTask)基于提供的客户及保单信息，创建并启动结构化识别任务。

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：CreateStructureTask。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	否	String	公共参数，本接口不需要传递此参数。
ServiceType	是	String	服务类型 Structured 仅结构化 Underwrite 结构化+核保 示例值：Structured
TaskInfos.N	是	Array of CreateStructureTaskInfo	创建任务时可以上传多个报告，后台生成多个识别子任务，子任务的详细信息 示例值：[{ "TaskType": "HealthReport", "CustomerId": "04
PolicyId	否	String	保单号 示例值：p0001
TriggerType	否	String	核保触发方式 Auto 自动 Manual 手动 示例值：Auto
InsuranceTypes.N	否	Array of String	险种，如果是体检报告类型，此参数是必填，类型说明如下： CriticalDiseaseInsurance:重疾险 LifeInsurance：寿险 AccidentInsurance：意外险 示例值：["CriticalDiseaseInsurance"]
CallbackUrl	否	String	回调地址，接收Post请求传送结果

3. 输出参数

参数名称	类型	描述
MainTaskId	String	创建的主任务号，用于查询结果 示例值：46313288149458944
RequestId	String	唯一请求 ID，每次请求都会返回。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 创建结构化任务接口示例

输入示例

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

{
"ServiceType": "Underwrite",
"PolicyId": "0406-01",
"TriggerType": "Auto",
"InsuranceTypes": [
"CriticalDiseaseInsurance",
"LifeInsurance",
"AccidentInsurance"
],
"TaskInfos": [
{
"TaskType": "HealthReport",
"CustomerId": "0406-01-1",
"CustomerName": "0406-01-1",
"FileList": [
"original_upload_dir/test.pdf"
],
"ImageList": [],
"Year": "2021"
}
]
}
```

输出示例

```
{
"Response": {
"RequestId": "22dfcc05-1ba1-49b4-a751-f5611cdb3420",
"MainTaskId": "37835597617481728"
}
}
```

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
AuthFailure	CAM签名/鉴权错误。
FailedOperation	操作失败。
InternalError	内部错误。
InvalidParameter	参数错误。
InvalidParameterValue	参数取值错误。
LimitExceeded	超过配额限制。
MissingParameter	缺少参数错误。
OperationDenied	操作被拒绝。

根据结构化任务ID创建核保任务

最近更新时间：2023-11-30 01:53:26

1. 接口描述

接口请求域名： cii.tencentcloudapi.com 。

本接口(CreateUnderwriteTaskById)用于根据结构化任务ID创建核保任务

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：CreateUnderwriteTaskById。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	否	String	公共参数，本接口不需要传递此参数。
MainTaskIds.N	是	Array of String	主任务ID数组， 示例值：["Mkjwdfso"]
CallbackUrl	否	String	回调地址，可不传（提供轮询机制）。 示例值：http://xxx.com/a/b/c

3. 输出参数

参数名称	类型	描述
UnderwriteTaskIds	Array of String	核保任务ID数据 示例值：["Ukjwssdfsos"]
RequestId	String	唯一请求 ID，每次请求都会返回。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 根据结构化任务ID创建核保任务示例

输入示例

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

```
{
"MainTaskIds": [
"Mkjwdfso"
]
}
```

输出示例

```
{
"Response": {
"RequestId": "lkjjl-asdf-er234sdf-sdfs",
"UnderwriteTaskIds": [
"Ukjwssdfsos"
]
}
}
```

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
AuthFailure	CAM签名/鉴权错误。

错误码	描述
FailedOperation	操作失败。
InternalError	内部错误。
InvalidParameter	参数错误。
InvalidParameterValue	参数取值错误。
LimitExceeded	超过配额限制。
MissingParameter	缺少参数错误。

结构化增量子任务

最近更新时间：2023-11-30 01:53:24

1. 接口描述

接口请求域名： cii.tencentcloudapi.com 。

如果主任务下的报告不满足需求，可以基于主任务批量添加子任务

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：AddSubStructureTasks。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	否	String	公共参数，本接口不需要传递此参数。
MainTaskId	是	String	主任务id 示例值：Mr595vgwutc0
TaskInfos.N	是	Array of CreateStructureTaskInfo	子任务信息数组

3. 输出参数

参数名称	类型	描述
SubTaskIds	Array of String	增量子任务id数组
RequestId	String	唯一请求 ID，每次请求都会返回。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 获取报告分类结果

输入示例

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

{
```

```
"MainTaskId": "Mstiljhgwglc",
"TaskInfos": [
{
"TaskType": "MedCheckReport",
"CustomerId": "1",
"CustomerName": "小明",
"FileList": [
"xxx.jpg",
"yyy.jpg"
]
}
]
}
```

输出示例

```
{
"Response": {
"RequestId": "22dfcc05-1ba1-49b4-a751-f5611cdb3420",
"SubTaskIds": [
"Sr595vgwutc1",
"Sr5fb7zin9xd"
]
}
}
```

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。
InternalError	内部错误。
InvalidParameter	参数错误。
InvalidParameterValue	参数取值错误。
MissingParameter	缺少参数错误。

查询结构化结果接口

最近更新时间：2023-11-30 01:53:22

1. 接口描述

接口请求域名： cii.tencentcloudapi.com 。

本接口(DescribeStructureResult)用于查询结构化结果接口

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：DescribeStructureResult。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	否	String	公共参数，本接口不需要传递此参数。
MainTaskId	是	String	创建任务时返回的主任务ID 示例值：Sjepg20lk7i9

3. 输出参数

参数名称	类型	描述
Status	Integer	结果状态： 0：返回成功 1：结果未生成 2：结果生成失败
Results	Array of StructureResultObject	结构化结果
MainTaskId	String	主任务ID
RequestId	String	唯一请求 ID，每次请求都会返回。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 查询结构化结果接口示例

输入示例

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

```
X-TC-Action: DescribeStructureResult
<公共请求参数>

{
"MainTaskId": "37835597617481728"
}
```

输出示例

```
{
"Response": {
"Status": 1,
"MainTaskId": "xx",
"Results": [
{
"TaskFiles": [
"xx"
],
"Code": 1,
"StructureResult": "xx",
"SubTaskId": "xx",
"TaskType": "xx"
}
],
"RequestId": "xx"
}
}
```

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
AuthFailure	CAM签名/鉴权错误。
FailedOperation	操作失败。
InternalError	内部错误。
InvalidParameter	参数错误。
InvalidParameterValue	参数取值错误。
LimitExceeded	超过配额限制。
MissingParameter	缺少参数错误。
OperationDenied	操作被拒绝。
RequestLimitExceeded	请求的次数超过了频率限制。

新建自动分类结构化任务

最近更新时间：2023-11-30 01:53:24

1. 接口描述

接口请求域名： cii.tencentcloudapi.com 。

本接口(CreateAutoClassifyStructureTask)基于提供的客户及保单信息，创建并启动结构化识别任务。

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：CreateAutoClassifyStructureTask。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	否	String	公共参数，本接口不需要传递此参数。
ServiceType	是	String	服务类型 Structured 仅结构化 Underwrite 结构化+核保 示例值：Structured
TaskInfos.N	是	Array of CreateAutoClassifyStructureTaskInfo	创建任务时可以上传多个报告，后台生成多个识别子任务，子任务的详细信息
PolicyId	否	String	保单号 示例值：p0001
TriggerType	否	String	核保触发方式 Auto 自动 Manual 手动 示例值：Auto
InsuranceTypes.N	否	Array of String	险种，如果是体检报告类型，此参数是必填，类型说明如下： CriticalDiseaseInsurance:重疾险 LifeInsurance：寿险 AccidentInsurance：意外险 示例值：["CriticalDiseaseInsurance"]
CallbackUrl	否	String	回调地址，接收Post请求传送结果

3. 输出参数

参数名称	类型	描述

参数名称	类型	描述
MainTaskId	String	创建的主任务号，用于查询结果 示例值：46313288149458944
RequestId	String	唯一请求 ID，每次请求都会返回。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 创建未分类结构化任务接口示例

输入示例

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

{
"ServiceType": "Underwrite",
"PolicyId": "0406-01",
"TriggerType": "Auto",
"InsuranceTypes": [
"CriticalDiseaseInsurance",
"LifeInsurance",
"AccidentInsurance"
],
"TaskInfos": [
{
"CustomerId": "0406-01-1",
"CustomerName": "0406-01-1",
"FileList": [
"original_upload_dir/test.pdf"
],
"ImageList": []
}
]
}
```

输出示例

```
{
"Response": {
"RequestId": "22dfcc05-1ba1-49b4-a751-f5611cdb3420",
"MainTaskId": "37835597617481728"
}
}
```

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。
InternalError	内部错误。
InvalidParameter	参数错误。
InvalidParameterValue	参数取值错误。
LimitExceeded	超过配额限制。
MissingParameter	缺少参数错误。
OperationDenied	操作被拒绝。

结构化对比相关接口
结构化对比查询

最近更新时间：2023-11-30 01:53:21

1. 接口描述

接口请求域名： cii.tencentcloudapi.com 。

结构化对比查询接口，对比结构化复核前后数据差异，查询识别正确率，召回率。

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：DescribeStructCompareData。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	否	String	公共参数，本接口不需要传递此参数。
MainTaskId	否	String	主任务号 示例值：fybnnac89vk
SubTaskId	否	String	子任务号 示例值：fybnnac89vl

3. 输出参数

参数名称	类型	描述
PolicyId	String	保单号 示例值：100001
MainTaskId	String	主任务号 示例值：fybnnac89vk
CustomerId	String	客户号 注意：此字段可能返回 null，表示取不到有效值。 示例值：100001
CustomerName	String	客户姓名 注意：此字段可能返回 null，表示取不到有效值。 示例值：张三
ReviewTime	String	复核时间 示例值：2020-12-19 18:43:37

参数名称	类型	描述
MachineResult	String	算法识别结果 示例值：{...}
ManualResult	String	人工复核结果 示例值：{...}
Metrics	CompareMetricsData	结构化对比指标数据
NewItems	String	新增项 示例值：{...}
ModifyItems	String	修改项 示例值：{...}
SubTaskId	String	子任务号 示例值：fybnnac89vl
AllTasks	Array of ReviewDataTaskInfo	所有的子任务
TaskType	String	任务类型 示例值：BUltrasoundReport
RequestId	String	唯一请求 ID，每次请求都会返回。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 结构化对比查询接口示例

输入示例

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

{
"MainTaskId": "fybnnac89vk",
"SubTaskId": ""
}
```

输出示例

```
{
"Response": {
"RequestId": "9317f6e4-6636-41a0-bf24-89ad9e4877f2",
"PolicyId": "100001",
"MainTaskId": "fybnnac89vk",
"SubTaskId": "fybnnac89vl",
"CustomerId": "100001",
"CustomerName": "张三",
"ReviewTime": "2020-12-19 18:43:37",
"MachineResult": "{...}",
```

```
"ManualResult": "{...}",
"NewItems": "",
"ModifyItems": "",
"Metrics": {
"ShortStructAccuracy": "95",
"ShortStructRecall": "11",
"LongStructAccuracy": "93",
"LongStructRecall": "14",
"LongContentAccuracy": "94",
"LongContentRecall": "14"
},
"AllTasks": [
{
"MainTaskId": "fybnnac89vk",
"SubTaskId": "fybnnac89vl",
"TaskName": "体检报告",
"TaskType": "HealthReport"
}
],
"TaskType": "BUltrasoundReport"
}
}
```


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。
InternalError	内部错误。
InvalidParameter	参数错误。

结构化复核差异查询

最近更新时间：2023-11-30 01:53:20

1. 接口描述

接口请求域名： cii.tencentcloudapi.com 。

结构化复核差异查询接口，对比结构化复核前后数据差异，返回差异的部分。

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：DescribeStructureDifference。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	否	String	公共参数，本接口不需要传递此参数。
MainTaskId	否	String	主任务号 示例值：fybnnac89vk
SubTaskId	否	String	子任务号 示例值：fybnnac89vl

3. 输出参数

参数名称	类型	描述
MainTaskId	String	主任务号 示例值：fybnnac89vk
Status	Integer	结果状态： 0：返回成功 1：结果未生成 2：结果生成失败 注意：此字段可能返回 null，表示取不到有效值。 示例值：0
Results	Array of PerStructDifference	差异的结果数组
RequestId	String	唯一请求 ID，每次请求都会返回。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 结构化对比查询接口示例

输入示例

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

{
"MainTaskId": "fybnnac89vk",
"SubTaskId": ""
}
```

输出示例


```
{
"Response": {
"MainTaskId": "fybnnac89vk",
"Status": 0,
"Results": [
{
"SubTaskId": "Snz0ypwht1xd",
"TaskType": "BUltrasoundReport",
"ModifyItems": [
{
"Path": "detailInfo/desc/organs/imageFeature/src",
"Machine": "回声",
"Manual": "回声均匀"
}
],
"NewItems": [
{
"Path": "detailInfo/summary/syms/attrs/性质",
"Value": "{\"__key\":\"f8ycfblrrh8\",\"desc\":\"性质\",\"indexChar\":-1,\"indexRow\":-1,\"positive\":\"\",\"src\":\"\",\"type\":\"\",\"value\":\"测试888888\"}"
}
],
"RemoveItems": [
{
"Path": "detailInfo/desc/organs/sizes",
"Value": "[{\"__key\":\"nz0yrpr2ux4\",\"desc\":\"\",\"indexChar\":130,\"indexRow\":0,\"numbers\":[29],\"src\":\"厚2.9CM\",\"type\":\"厚\",\"unit\":\"MM\",\"value\":\"厚2.9CM\"}]"
}
]
}
],
"RequestId": "6e2f2787-1851-4bf1-8477-08c93b6ee652"
}
}
```

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。
InternalError	内部错误。
InvalidParameter	参数错误。

核保相关接口
查询机器核保任务数据

最近更新时间：2023-11-30 01:53:25

1. 接口描述

接口请求域名： cii.tencentcloudapi.com 。

本接口(DescribeMachineUnderwrite)用于查询机器核保任务数据

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：DescribeMachineUnderwrite。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	否	String	公共参数，本接口不需要传递此参数。
UnderwriteTaskId	是	String	核保任务ID 示例值：u-task-1

3. 输出参数

参数名称	类型	描述
Uin	String	腾讯云主账号ID 示例值：100000000123
SubAccountUin	String	操作人子账户ID 示例值：100000000123
PolicyId	String	保单ID 示例值：id-123456
MainTaskId	String	主任务ID 示例值：Malksjdkjl234
UnderwriteTaskId	String	核保任务ID 示例值：U23slkd
Status	Integer	结果状态： 0：返回成功 1：结果未生成 2：结果生成失败 示例值：0

参数名称	类型	描述
UnderwriteResults	Array of MachineUnderwriteOutput	机器核保结果
RequestId	String	唯一请求 ID，每次请求都会返回。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 查询机器核保任务数据示例

输入示例

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

{
"UnderwriteTaskId": "3456234"
}
```

输出示例

```
{
"Response": {
"SubAccountUin": "xx",
"Status": 0,
"RequestId": "xx",
"Uin": "xx",
"UnderwriteResults": [
{
"CustomerId": "xx",
"CustomerName": "xx",
"Results": [
{
"InsuranceType": "xx",
"Result": [
{
"Laboratory": [
{
"Name": "xx",
"Value": "xx",
"Result": "xx"
}
],
"Explanation": [
{
```

```
"Name": "xx",
"Value": "xx",
"Result": "xx"
}
],
"Conclusion": "xx",
"Disease": [
{
"Name": "xx",
"Value": "xx",
"Result": "xx"
}
],
"Title": "xx"
}
]
}
]
}
],
"PolicyId": "xx",
"MainTaskId": "xx",
"UnderwriteTaskId": "xx"
}
}
```


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。
InternalError	内部错误。
InvalidParameter	参数错误。
InvalidParameterValue	参数取值错误。
LimitExceeded	超过配额限制。

查询核保任务数据

最近更新时间：2023-11-30 01:53:25

1. 接口描述

接口请求域名： cii.tencentcloudapi.com 。

本接口(DescribeUnderwriteTask)用于查询核保任务结果

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer 点击调试

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

2. 输入参数

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

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：DescribeUnderwriteTask。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	否	String	公共参数，本接口不需要传递此参数。
UnderwriteTaskId	否	String	任务ID 示例值：u-task-1

3. 输出参数

参数名称	类型	描述
Uin	String	腾讯云主账号ID 示例值：100000000123
SubAccountUin	String	操作人子账户ID 示例值：100000000123
PolicyId	String	保单ID 示例值：id-123456
MainTaskId	String	主任务ID 示例值：Malksjdkjl234
UnderwriteTaskId	String	核保任务ID 示例值：U23slkd
Status	Integer	结果状态： 0：返回成功 1：结果未生成 2：结果生成失败 示例值：0
UnderwriteResults	Array of UnderwriteOutput	核保结果

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

4. 示例

示例1 查询核保任务数据示例

输入示例

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

{
"UnderwriteTaskId": "3456234"
}
```

输出示例

```
{
"Response": {
"SubAccountUin": "xx",
"Status": 0,
"RequestId": "xx",
"Uin": "xx",
"UnderwriteResults": [
{
"ManualDetail": [
{
"Explanation": "xx",
"Type": "xx",
"Conclusion": "xx"
}
],
"ReviewTime": "xx",
"Results": [
{
"InsuranceType": "xx",
"Result": [
{
"Laboratory": [
{
"Name": "xx",
"Value": "xx",
"Result": "xx"
}
],
```

```
"Explanation": [
{
"Name": "xx",
"Value": "xx",
"Result": "xx"
}
],
"Conclusion": "xx",
"Disease": [
{
"Name": "xx",
"Value": "xx",
"Result": "xx"
}
],
"Title": "xx"
}
]
}
],
"CustomerName": "xx",
"CustomerId": "xx"
}
],
"PolicyId": "xx",
"MainTaskId": "xx",
"UnderwriteTaskId": "xx"
}
}
```


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。
InternalError	内部错误。
InvalidParameter	参数错误。
InvalidParameterValue	参数取值错误。
LimitExceeded	超过配额限制。
MissingParameter	缺少参数错误。

报告自动分类

最近更新时间：2023-11-30 01:53:22

1. 接口描述

接口请求域名： cii.tencentcloudapi.com 。

辅助用户对批量报告自动分类

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：DescribeReportClassify。
Version	是	String	公共参数，本接口取值：2021-04-08。
Region	否	String	公共参数，本接口不需要传递此参数。
ServiceType	是	String	服务类型 Structured 仅结构化 Underwrite 结构化+核保 示例值：Structured
FileList.N	是	Array of String	文件地址数组 示例值：["xxxx/xx.jpg", "xxxx/xx.jpg"]

3. 输出参数

参数名称	类型	描述
Reports	Array of ClassifiedReports	报告分类结果
RequestId	String	唯一请求 ID，每次请求都会返回。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 获取报告分类结果

输入示例

```
POST / HTTP/1.1
Host: cii.tencentcloudapi.com
Content-Type: application/json
X-TC-Action: DescribeReportClassify
```

```
<公共请求参数>

{
"ServiceType": "Structured",
"FileList": [
"700000198392/original_upload_dir/700000198392_b0a3da0c-af75-464f-b202-f49c7127c012.pdf",
"700000198392/original_upload_dir/700000198392_6622637e-4da0-11ec-b930-525400f962860_rh95vodxdkw.jpg"
]
}
```

输出示例

```
{
"Response": {
"RequestId": "22dfcc05-1ba1-49b4-a751-f5611cdb3420",
"Reports": [
{
"ReportType": "HealthReport",
"FileList": [
"700000198392/original_upload_dir/700000198392_b0a3da0c-af75-464f-b202-f49c7127c012.pdf"
]
},
{
"ReportType": "MedCheckReport",
"FileList": [
"700000198392/original_upload_dir/700000198392_6622637e-4da0-11ec-b930-525400f962860_rh95vodxdkw.jpg"
]
}
]
}
}
```

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
AuthFailure	CAM签名/鉴权错误。
FailedOperation	操作失败。
InternalError	内部错误。
InvalidParameter	参数错误。
InvalidParameterValue	参数取值错误。
MissingParameter	缺少参数错误。
OperationDenied	操作被拒绝。

基础通用结构

最近更新时间：2022-02-14 08:07:58

基础通用结构 BasicInfo

名称	类型	描述
depart	string	科室
hospital	string	医院
reportTime	string	报告时间
reportType	string	报告类型
outpatientNum	string	诊号
inHospitalNum	string	住院号
checkNum	string	检查号
imageNum	string	影像号
radiationNum	string	放射号
pathologyNum	string	病理号
inHospitalDate	string	入院时间
outHospitalDate	string	出院时间
inspectTime	string	检查时间
inHospitalDay	string	住院天数
checkMethod	string	检查方法
diagnose	string	临床诊断
inHospitalDayNum	string	住院次数
bedNum	string	床号

患者信息 PatientInfo

名称	类型	描述
name	string	姓名
age	string	年龄
sex	string	性别
phone	string	联系方式
address	string	现住址

名称	类型	描述
idCard	string	身份证
healthCardNo	string	健康卡号
socialSecurityCardNo	string	社保卡号
birthday	string	出生日期
ethnicity	string	族
nationality	string	国籍
married	string	婚姻状况
profession	string	职业
educationBackground	string	教育程度
birthPlace	string	籍贯
medicalInsuranceType	string	医保类型

响应头 RspHead

名称	类型	描述
code	int32	响应消息码
message	string	响应消息

部位信息 NormPart

名称	类型	描述
desc	string	描述
partDirection	string	部位方向
partValue	string	部位
tissueDirection	string	组织方向
tissueValue	string	组织
upper	string	上方
value	string	部位内容
src	string	原文内容
indexRow	int32	原文在描述文字的字符位置索引
indexChar	int32	原文在描述文字的字符位置索引

大小信息 NormSize

名称	类型	描述
desc	string	描述
type	string	类型
numbers	Array of float	大小数值
unit	string	单位
value	string	大小内容
src	string	原文内容
indexRow	int32	原文在描述文字的行号位置索引
indexChar	int32	原文在描述文字的行号位置索引

通用块信息 BlockInfo

名称	类型	描述
desc	string	描述
positive	string	肯定信息
type	string	类型
value	string	提取信息
src	string	原文内容
indexRow	int32	原文在描述文字的行号位置索引
indexChar	int32	原文在描述文字的行号位置索引

诊断建议 Advice

名称	类型	描述
text	string	建议文本

诊断证明 DiagCertItem

名称	类型	描述
text	string	诊断文本
type	string	诊断类型

名称	类型	描述
value	Array of string	提取信息

检查报告结构说明

最近更新时间：2022-02-16 08:03:26

总体返回结构说明

名称	类型	描述
basicInfo	BasicInfo	基础信息
patientInfo	PatientInfo	患者信息
rspHead	RspHead	响应头
check	Check	检查报告

检查报告 Check

名称	类型	描述
desc	Desc	描述说明
summary	Summary	描述说明

报告描述 Desc

名称	类型	描述
text	string	文本内容
tubers	Array of Tuber	器官
organs	Array of Organ	组织

检查结论 Summary

名称	类型	描述
syms	Array of Sym	病变结论信息
text	string	文本信息

结节 Tuber

名称	类型	描述
part	NormPart	部位
multiple	Multiple	多发
sizes	Array of NormSize	大小部位信息
type	BlockInfo	类型信息
edge	BlockInfo	边缘信息
shape	BlockInfo	形状
cdfi	BlockInfo	血流cdfi
calcification	BlockInfo	钙化信息
enhancement	BlockInfo	强化信息
aspectRatio	AspectRatio	纵横比
rearEcho	BlockInfo	后方回声信息
elastic	Elastic	质地弹性
operation	BlockInfo	手术情况
lympEnlargement	BlockInfo	淋巴结信息
activity	BlockInfo	活动度
envelope	BlockInfo	包膜信息
skinMedulla	BlockInfo	皮髓质信息
lympDoor	BlockInfo	淋巴　信息
transparent	BlockInfo	透声度
shapeAttr	BlockInfo	形态
trend	BlockInfo	变化趋势
index	Array of int32	范围
sizeStatus	BlockInfo	大小状态
innerEchoDistribution	BlockInfo	内部回声分布
innerEchoType	Array of BlockInfo	内部回声类型
outline	BlockInfo	轮廓
structure	BlockInfo	结构
density	BlockInfo	密度
vas	BlockInfo	血管
cysticwall	BlockInfo	囊壁
capsule	BlockInfo	被膜
isthmusThicknese	NormSize	峡部厚度

名称	类型	描述
mriAdc	BlockInfo	mriAdc
mriDwi	BlockInfo	mriDwi
mriT1	BlockInfo	mriT1
mriT2	BlockInfo	mriT2
ctHu	BlockInfo	CT_HU
suvmax	BlockInfo	SUVMAX
metabolism	BlockInfo	代谢情况
radioactiveUptake	BlockInfo	放射性摄取
src	string	原文内容
imageFeature	BlockInfo	影像特征

器官 Organ

名称	类型	描述
gland	BlockInfo	腺体信息
part	NormPart	部位
sizes	Array of NormSize	大小部位信息
shape	BlockInfo	形状
thickness	BlockInfo	厚度
transparent	BlockInfo	原echo内部回声信息
envelope	BlockInfo	包膜信息
edge	BlockInfo	边缘信息
cdfi	BlockInfo	血流cdfi
index	Array of int32	范围
shapeAttr	BlockInfo	形态
symDesc	BlockInfo	描述信息
sizeStatus	BlockInfo	大小状态
outline	BlockInfo	轮廓
structure	BlockInfo	结构
density	BlockInfo	密度
vas	BlockInfo	血管
cysticwall	BlockInfo	囊壁

名称	类型	描述
capsule	BlockInfo	被膜
isthmusThicknese	NormSize	峡部厚度
innerEchoDistribution	BlockInfo	内部回声分布
mriAdc	BlockInfo	mriAdc
mriDwi	BlockInfo	mriDwi
mriT1	BlockInfo	mriT1
mriT2	BlockInfo	mriT2
ctHu	BlockInfo	CT_HU
suvmax	BlockInfo	SUVMAX
metabolism	BlockInfo	代谢情况
radioactiveUptake	BlockInfo	放射性摄取
innerEchoType	Array of BlockInfo	内部回声类型
lympEnlargement	BlockInfo	淋巴结信息
src	string	原文内容
imageFeature	BlockInfo	影像特征

病变 Sym

名称	类型	描述
grade	BlockInfo	组织学分级
part	NormPart	部位
index	Array of int32	范围
sym	BlockInfo	组织
attrs	Array of BlockInfo	块信息
src	string	原文内容

纵横比 AspectRatio

名称	类型	描述
desc	string	描述说明
value	string	提取信息
relation	string	关系

名称	类型	描述
number	float	数字
src	string	原文内容
indexRow	int32	原文在描述文字的行号位置索引
indexChar	int32	原文在描述文字的行号位置索引

多发 Multiple

名称	类型	描述
src	string	原文内容
value	string	部位内容
count	int32	次数
desc	string	描述内容
indexRow	int32	原文在描述文字的行号位置索引
indexChar	int32	原文在描述文字的行号位置索引

弹性质地 Elastic

名称	类型	描述
desc	string	描述
value	string	提取信息
score	string	评分
src	string	原文内容
indexRow	int32	原文在描述文字的行号位置索引
indexChar	int32	原文在描述文字的行号位置索引

病理报告结构说明

最近更新时间：2022-02-16 08:03:26

总体返回结构说明

名称	类型	描述
basicInfo	BasicInfo	基础信息
patientInfo	PatientInfo	患者信息
rspHead	RspHead	响应头
pathology	Pathology	病理报告

病理报告 Pathology

名称	类型	描述
sampleType	BasicInfo	标本类型
summaryText	string	结论文本
descText	string	描述文本
cancer	Cancer	癌症
invasives	Array of Invasive	侵犯
lymphs	Array of Lymph	淋巴
ihc	Array of Ihc	ihc
pathologicalReportType	BasicInfo	病理报告类型
pTNM	BasicInfo	pTNM
reportText	string	报告原文
infiltration_depth	BasicInfo	浸润深度

癌症信息 Cancer

名称	类型	描述
part	NormPart	部位
sizes	Array of NormSize	大小
histologyType	BasicInfo	组织学类型
histologyLevel	BasicInfo	组织学分级

侵犯 Invasive

名称	类型	描述
src	string	原文内容
part	NormPart	部位
positive	string	阳性
indexRow	int32	原文在描述文字的行号位置索引
indexChar	int32	原文在描述文字的行号位置索引

淋巴 Lymph

名称	类型	描述
transferNum	string	转移数值
src	string	原文内容
indexRow	int32	原文在描述文字的行号位置索引
indexChar	int32	原文在描述文字的行号位置索引
part	NormPart	部位
total	string	数量

免疫组化 Ihc

名称	类型	描述
value	Ihc.Value	ihc值
src	string	原文内容
name	string	ihc名称
indexRow	int32	原文在描述文字的行号位置索引
indexChar	int32	原文在描述文字的行号位置索引

免疫组化值 Ihc.Value

名称	类型	描述
grade	string	等级
percent	Array of string	百分比

名称	类型	描述
positive	string	阳性

诊断证明结构说明

最近更新时间：2022-02-16 08:03:26

总体返回结构说明

名称	类型	描述
basicInfo	BasicInfo	基础信息
patientInfo	PatientInfo	患者信息
rspHead	RspHead	响应头
diagCert	DiagCert	诊断证明

诊断证明 DiagCert

名称	类型	描述
advice	Advice	建议和医嘱
diagnosis	Array of DiagCertItem	诊断证明

出入院报告结构说明

最近更新时间：2022-02-16 08:03:27

总体返回结构说明

名称	类型	描述
basicInfo	BasicInfo	基础信息
patientInfo	PatientInfo	患者信息
rspHead	RspHead	响应头
medDoc	MedDoc	出入院报告

出入院报告 MedDoc

名称	类型	描述
advice	Advice	建议和医嘱
diagnosis	Array of DiagCertItem	诊断证明
disease	Disease	疾病历史
personal	Personal	个人过往史
obsterical	Obsterical	婚育史
family	Family	家族史
menstrual	Menstrual	月经史
treatmentRecord	TreatmentRecord	诊疗信息
bodyExamination	BodyExamination	体格检查
surgery	SurgeryInfo	手术信息

疾病史 Disease

名称	类型	描述
mainDisease	string	主病史
allergy	string	过敏史
infect	string	传染疾病史
operation	string	手术外伤史
transfusion	string	输血史

个人过往史 Personal

名称	类型	描述
birthPlace	string	出生地
livePlace	string	生活所在地
job	string	工作
smoke	string	吸烟史
alcoholic	string	饮酒史

婚育史 Obsterical

名称	类型	描述
marriage	string	婚姻史
fertility	string	生育史

家族史 Family

名称	类型	描述
relative	string	家属成员
relativeCancer	string	家族肿瘤史
genetic	string	遗传史

月经史 Menstrual

名称	类型	描述
lastMenstrualPeriod	string	最后月经时间
menstrualFlow	string	经量
menarcheAge	string	月经初潮年龄
menstruationOrNot	string	是否来月经
menstrualCycles	string	月经周期
menstrualPeriod	string	行经天数

诊疗信息 TreatmentRecord

名称	类型	描述
admissionCondition	string	入院诊断
chiefComplaint	string	主诉
diseasePresent	string	现病史
symptomsAndSigns	string	主要症状体征
auxiliaryExamination	string	辅助检查
bodyExamination	string	体格检查
specialistExamination	string	专科检查
mentalExamination	string	精神检查
checkRecord	string	检查记录
inspectResult	string	化验结果
incisionHealing	string	切口愈合情况
treatmentSuggestion	string	处理意
followUpRequirements	string	诊随访要求
checkAndTreatmentProcess	string	诊疗经过
surgeryCondition	string	手术经过
conditionChanges	string	入院情况
dischargeCondition	string	出院情况
pTNM	string	pTNM
pTNMM	string	pTNMM
pTNMN	string	pTNMN
pTNMT	string	pTNMT
ECOG	string	ECOG
NRS	string	NRS
KPS	string	KPS
clinicalStaging	string	临床分期
deathDate	string	死亡日期
relapseDate	string	复发日期
observationDays	string	观测天数
admissionDiagnosis	string	入院诊断
dischargeDiagnosis	string	出院诊断
diseaseDiagnosis	string	病理诊断

名称	类型	描述
dischargeTraditionalDiagnosis	string	出院中医诊断
dischargeModernDiagnosis	string	出院　医诊断
admissionTraditionalDiagnosis	string	入院中医诊断
admissionModernDiagnosis	string	入院　医诊断
dischargeInstructions	string	出院医嘱
pulmonaryArterySystolicPressure	string	肺动脉收缩压
BCLC	string	BCLC分期

体格检查 BodyExamination

名称	类型	描述
bodyTemperature	string	体温
pulse	string	脉搏
breathe	string	呼吸
bloodPressure	string	血压

手术信息 SurgeryInfo

名称	类型	描述
surgeryName	string	手术名称
surgeryPart	string	手术部位

病案首页结构说明

最近更新时间：2022-02-16 08:03:27

总体返回结构说明

名称	类型	描述
basicInfo	BasicInfo	基础信息
patientInfo	PatientInfo	患者信息
rspHead	RspHead	响应头
firstPage	FirstPage	病案首页

病案首页 FirstPage

名称	类型	描述
dischargeDiagnosis	Array of DischargeDiagnosis	出院诊断
pathologicDiagnosis	FirstPageAttr	病理诊断
clinicDiagnosis	FirstPageAttr	（急)诊诊断
damagePoi	FirstPageAttr	损伤、中毒的外部原因

出院诊断 DischargeDiagnosis

名称	类型	描述
tableIndex	int32	表格位置
outDiagnosis	BasicInfo	出院诊断
diseaseCode	BasicInfo	疾病编码
inStatus	BasicInfo	入院情况
outStatus	BasicInfo	出院情况

FirstPageAttr

名称	类型	描述
code	string	诊断编码
indexRow	int32	原文在描述文字的行号位置索引

名称	类型	描述
indexChar	int32	原文在描述文字的行号位置索引
name	string	属性描述
src	string	原文内容
value	string	提取信息

检验报告结构说明

最近更新时间：2022-02-16 08:03:27

总体返回结构说明

名称	类型	描述
basicInfo	BasicInfo	基础信息
patientInfo	PatientInfo	患者信息
rspHead	RspHead	响应头
laboratoryReport	LaboratoryReport	检验报告

检验报告 LaboratoryReport

名称	类型	描述
indicators	Array of IndicatorItem	检验报告单

检验报告单 IndicatorItem

名称	类型	描述
code	string	英文缩写
scode	string	标准缩写
name	string	项目名称
sname	string	标准名
result	string	结果
unit	string	单位
range	string	参考范围
arrow	string	上下箭头
normal	bool	结果是否异常

手术记录结构说明

最近更新时间：2022-02-16 08:03:26

总体返回结构说明

名称	类型	描述
basicInfo	BasicInfo	基础信息
patientInfo	PatientInfo	患者信息
rspHead	RspHead	响应头
surgery	Surgery	手术记录

手术记录 Surgery

名称	类型	描述
surgeryHistory	SurgeryHistory	手术史

手术史 SurgeryHistory

名称	类型	描述
surgeryName	SurgeryAttr	手术名称
surgeryDate	SurgeryAttr	手术时间
preoperativePathology	SurgeryAttr	术前诊断
intraoperativePathology	SurgeryAttr	术中诊断
postoperativePathology	SurgeryAttr	术后诊断
dischargeDiagnosis	SurgeryAttr	出院诊断

手术记录属性 SurgeryAttr

名称	类型	描述
name	string	属性
value	string	值

体检报告结构说明

最近更新时间：2022-03-29 08:06:30

总体返回结构说明

名称	类型	描述
IsOneself	bool	是否为该参保人
HealthReportMain	HealthReport	体检报告主体
HealthNotification	Array of HRValue	健康告知书
AccompanyInspection	Array of HRValue	陪检告知书
ImageList	Array of ImageItem	图片
ExceptionSet	Array of ExceptionSet	异常项

体检报告 HealthReport

名称	类型	描述
BasicInfo	Array of HRValue	基本信息
Functions	Array of HRValue	功能项目
PhysicalExamination	Array of HRItemValueSet	物理检查
LabExamination	Array of HRItemValueSet	实验室检查
ImageExamination	Array of HRImageItemValueSet	影像学检查
MedicalExamination	Array of HRItemValueSet	其他医技检查
Others	Array of HRItemValue	其他
ExceptionSet	Array of HRValue	异常汇总

HRValue

名称	类型	描述
KeyName	string	字段名称
PicId	string	图片ID
Value	string	值
NormalValue	string	标准值
Coords	Coordinate	坐标

名称	类型	描述
QualityConf	float32	置信度
Id	int	其他

图片 ImageItem

名称	类型	描述
PicId	string	图片ID
Pic64	string	图片Base64内容
PicType	string	图片类型

HRItemValue

名称	类型	描述
ItemName	string	检查项名称
ItemFlag	string	项目标记：是新增还是修改等
StandardFlag	bool	标准项标记：是否展示标准项
PicId	string	图片ID
OriginName	HRValue	原名
ItemValue	HRValue	值
IsPositive	HRValue	阴阳性
Ranges	HRValue	参考范围
Unit	HRValue	单位
StandardUnit	string	标准单位
StandardRange	string	标准参考范围

HRImageItemValue

名称	类型	描述
ItemName	string	检查项名称
ItemFlag	string	项目标记：是新增还是修改等
StandardFlag	bool	标准项标记：是否展示标准项
PicId	string	图片ID

名称	类型	描述
OriginName	HRValue	原名
ItemValue	HRValue	值
IsPositive	HRValue	阴阳性
Ranges	HRValue	参考范围
Unit	HRValue	单位
StructureResult	MedStructure	结构化
ExceptionConclusion	HRValue	一次总结
Conclusion	HRValue	结论
StandardUnit	string	标准单位
StandardRange	string	标准参考范围

MedStructure

名称	类型	描述
Cdfi	HRValue	CDFI
Organ	Array of HRValue	脏器
OccupyingLesion	Array of HRValue	占位病变

HRImageItemValueSet

名称	类型	描述
ItemSetName	string	具体检查项目集的名字,例如内科、外科
ImageItemValues	Array of HRImageItemValue	检查项目集内的具体检查项

HRItemValueSet

名称	类型	描述
ItemSetName	string	具体检查项目集的名字,例如内科、外科
ImageItemValues	Array of HRItemValue	检查项目集内的具体检查项

Coordinate

名称	类型	描述
X	int	x坐标
Y	int	y坐标
Width	int	宽
Height	int	高

ExceptionSet

名称	类型	描述
Path	string	例如：“体检报告/实验室检查/肿瘤标志物”
ExceptionValues	ExceptionValues	异常项信息

ExceptionValues

名称	类型	描述
Key	string	标准名称
Value	string	测量值
Name	string	原名
Flag	string	阴阳性（+/-）
Unit	string	单位
Range	string	参考范围

数据结构

最近更新时间：2023-08-17 01:49:28

ClassifiedReports

报告分类结果

被如下接口引用：DescribeReportClassify。

名称	类型	描述
ReportType	String	报告类型 示例值：AdmissionReport
FileList	Array of String	文件列表 示例值：["xxxx/xx.jpg", "xxxx/xx.jpg"]

ClassifyInfo

报告分类信息

被如下接口引用：DescribeStructureTaskResult。

名称	类型	描述
FirstClass	String	一级分类 示例值：检查报告
SecondClass	String	二级分类 示例值：超声检查
ThirdClass	String	三级分类 示例值：超声检查
FirstClassId	Integer	一级分类序号 示例值：12
SecondClassId	Integer	二级分类序号 示例值：345
ThirdClassId	Integer	三级分类序号 示例值：345

CompareMetricsData

结构化对比指标（准确率/召回率）数据

被如下接口引用：DescribeStructCompareData。

名称	类型	描述
ShortStructAccuracy	String	短文准确率 注意：此字段可能返回 null，表示取不到有效值。 示例值：94
ShortStructRecall	String	短文召回率 注意：此字段可能返回 null，表示取不到有效值。 示例值：91

名称	类型	描述
LongStructAccuracy	String	长文结构化准确率 注意：此字段可能返回 null，表示取不到有效值。 示例值：97
LongStructRecall	String	长文结构化召回率 注意：此字段可能返回 null，表示取不到有效值。 示例值：93
LongContentAccuracy	String	长文提取准确率 注意：此字段可能返回 null，表示取不到有效值。 示例值：93
LongContentRecall	String	长文提取召回率 注意：此字段可能返回 null，表示取不到有效值。 示例值：91

CreateAutoClassifyStructureTaskInfo

创建自动分类的结构化任务子任务信息

被如下接口引用：CreateAutoClassifyStructureTask。

名称	类型	必选	描述
FileList	Array of String	是	报告文件上传的地址列表，需按顺序排列。如果使用ImageList参数，置为空数组即可 示例值：["original_upload_dir/100000013512_e75d5c6e-7b90-4ff6-b890-483"]
CustomerId	String	否	客户号 示例值：C0001
CustomerName	String	否	客户姓名 示例值：测试客户
ImageList	Array of String	否	报告上传的图片内容数组，图片内容采用base64编码，需按顺序排列 示例值：["9xkKYHNKWDxxxxx"]

CreateStructureTaskInfo

创建结构化任务子任务信息

被如下接口引用：AddSubStructureTasks, CreateStructureTask。

名称	类型	必选	描述
TaskType	String	是	任务类型:HealthReport(体检报告); BUltraReport(B超报告);MedCheckReport(检查报告);LaboratoryReport(检验报告); PathologyReport(病理报告);AdmissionReport(入院记录);DischargeReport(出院记录); DischargeSummary(出院小结);DiagnosisReport(诊断证明); MedicalRecordFront(病案首页);OperationReport(手术记录);OutpatientMedicalRecord(门诊病历) 示例值：HealthReport
FileList	Array of String	是	报告文件上传的地址列表，需按顺序排列。如果使用ImageList参数，置为空数组即可 示例值：["original_upload_dir/100000013512_e75d5c6e-7b90-4ff6-b890-483"]
CustomerId	String	否	客户号 示例值：C0001

名称	类型	必选	描述
CustomerName	String	否	客户姓名 示例值：测试客户
ImageList	Array of String	否	报告上传的图片内容数组，图片内容采用base64编码，需按顺序排列 示例值：["9xkKYHNKWDxxxxx"]
Year	String	否	报告年份 示例值：2021

InsuranceResult

包含险种的各个核保结论

被如下接口引用：DescribeMachineUnderwrite, DescribeUnderwriteTask。

名称	类型	描述
InsuranceType	String	险种:CriticalDiseaseInsurance(重疾险);LifeInsurance(寿险);AccidentInsurance(意外险);MedicalInsurance(医疗险) 示例值：CriticalDiseaseInsurance
Result	Array of MachinePredict	对应险种的机器核保结果

Location

位置信息

被如下接口引用：DescribeStructureResult。

名称	类型	描述
Points	Array of Point	位置信息 示例值：[{"x"=1,"y" =1}]

MachinePredict

机器核保预测结果

被如下接口引用：DescribeMachineUnderwrite, DescribeUnderwriteTask。

名称	类型	描述
Title	String	核保引擎名称 示例值：AI决策
Conclusion	String	核保结论：加费、承保、拒保、延期、除外、加费+除外 示例值：拒保
Explanation	Array of UnderwriteItem	AI决策树解释
Disease	Array of UnderwriteItem	疾病指标
Laboratory	Array of UnderwriteItem	检查异常

MachineUnderwriteOutput

机器核保输出

被如下接口引用：DescribeMachineUnderwrite。

名称	类型	描述
CustomerId	String	客户号 示例值：123
CustomerName	String	客户姓名 示例值：xxx
Results	Array of InsuranceResult	各个险种的结果

OcrRecognise

Ocr识别结果

被如下接口引用：DescribeStructureResult。

名称	类型	描述
OriginalField	String	原文字段 示例值：“”
Value	String	识别结果 示例值：“”
Confidence	Float	置信度 示例值：0
Location	Location	位置信息 示例值：{}
Field	String	字段名 示例值：“”

PerStructDifference

复核差异接口的每一份报告的差异结果

被如下接口引用：DescribeStructureDifference。

名称	类型	描述
SubTaskId	String	子任务ID 示例值：dsflnnlau
TaskType	String	任务类型:HealthReport(体检报告); BUltraReport(B超报告);MedCheckReport(检查报告);LaboratoryReport(检验报告); PathologyReport(病理报告);AdmissionReport(入院记录);DischargeReport(出院记录); DischargeSummary(出院小结);DiagnosisReport(诊断证明); MedicalRecordFront(病案首页);OperationReport(手术记录);OutpatientMedicalRecord(门诊病历) 示例值：HealthReport
ModifyItems	Array of StructureModifyItem	修改的项
NewItems	Array of StructureOneItem	新增的项

名称	类型	描述
RemoveItems	Array of StructureOneItem	删除的项

Point

点信息

被如下接口引用：DescribeStructureResult。

名称	类型	描述
XCoordinate	Integer	x坐标 示例值：1
YCoordinate	Integer	y坐标 示例值：1
Page	Integer	页码 示例值：1

ResultObject

用于返回结构化任务结果

被如下接口引用：DescribeStructureTaskResult。

名称	类型	描述
Quality	Float	图片质量分 示例值：0.87
StructureResult	String	由结构化算法结构化json转换的字符串，具体协议参见算法结构化结果协议 示例值：{...}
ReportType	Array of ClassifyInfo	报告分类信息 注意：此字段可能返回 null，表示取不到有效值。

ReviewDataTaskInfo

人工复核数据的子任务信息

被如下接口引用：DescribeStructCompareData。

名称	类型	描述
MainTaskId	String	主任务号 示例值：fybnnac89vk
SubTaskId	String	子任务号 示例值：fybnnac89vl
TaskName	String	任务名 示例值：体检报告

名称	类型	描述
TaskType	String	任务类型:HealthReport(体检报告); BUltraReport(B超报告);MedCheckReport(检查报告);LaboratoryReport(检验报告); PathologyReport(病理报告);AdmissionReport(入院记录);DischargeReport(出院记录); DischargeSummary(出院小结);DiagnosisReport(诊断证明); MedicalRecordFront(病案首页);OperationReport(手术记录);OutpatientMedicalRecord(门诊病历) 示例值：HealthReport

StructureModifyItem

结构化复核差异接口的修改的项

被如下接口引用：DescribeStructureDifference。

名称	类型	描述
Path	String	修改的字段的路径 注意：此字段可能返回 null，表示取不到有效值。 示例值：detailInfo/summary/syms/attrs/性质
Machine	String	机器结果的值 注意：此字段可能返回 null，表示取不到有效值。 示例值：100cm
Manual	String	人工结果的值 注意：此字段可能返回 null，表示取不到有效值。 示例值：120cm

StructureOneItem

复核差异接口的新增或者删除的项

被如下接口引用：DescribeStructureDifference。

名称	类型	描述
Path	String	新字段的路径 注意：此字段可能返回 null，表示取不到有效值。 示例值：detailInfo/summary/syms/attrs/性质
Value	String	字段的值 注意：此字段可能返回 null，表示取不到有效值。 示例值：10cm

StructureResultObject

结构化结果

被如下接口引用：DescribeStructureResult。

名称	类型	描述
Code	Integer	0表示正常返回；1代表结果未生成；2代表任务执行失败 示例值：0

名称	类型	描述
TaskType	String	报告类型:HealthReport(体检报告); BUltraReport(B超报告);MedCheckReport(检查报告);LaboratoryReport(检验报告); PathologyReport(病理报告);AdmissionReport(入院记录);DischargeReport(出院记录); DischargeSummary(出院小结);DiagnosisReport(诊断证明); MedicalRecordFront(病案首页);OperationReport(手术记录);OutpatientMedicalRecord(门诊病历) 示例值：HealthReport
StructureResult	String	结构化结果
SubTaskId	String	子任务ID
TaskFiles	Array of String	任务文件列表
ResultFields	Array of OcrRecognise	结构化字段结果数组 注意：此字段可能返回 null，表示取不到有效值。 示例值：{}

UnderwriteConclusion

核保结论 机器结论和人工结论统一数据结构

被如下接口引用：DescribeUnderwriteTask。

名称	类型	描述
Type	String	类型 示例值：CriticalDiseaseInsurance
Conclusion	String	结论 示例值：Accept
Explanation	String	解释 示例值："{}"

UnderwriteItem

机器核保结论子项

被如下接口引用：DescribeMachineUnderwrite, DescribeUnderwriteTask。

名称	类型	描述
Name	String	字段名 示例值：心率
Result	String	结果 示例值：80
Value	String	风险值或者说明 示例值：+0
Range	String	参考范围 注意：此字段可能返回 null，表示取不到有效值。 示例值：1-2
ReportDate	Array of String	报告时间 注意：此字段可能返回 null，表示取不到有效值。 示例值：[]

名称	类型	描述
FileType	String	文件类型 注意：此字段可能返回 null，表示取不到有效值。 示例值：healthReport
InspectProject	String	检查项目 注意：此字段可能返回 null，表示取不到有效值。 示例值：血糖
Unit	String	单位 注意：此字段可能返回 null，表示取不到有效值。 示例值：""
OriginName	String	原名 注意：此字段可能返回 null，表示取不到有效值。 示例值：空腹血葡萄糖
YinYang	String	阴阳性 注意：此字段可能返回 null，表示取不到有效值。 示例值：+

UnderwriteOutput

核保结果输出

被如下接口引用：DescribeUnderwriteTask。

名称	类型	描述
CustomerId	String	客户ID 示例值：user123
CustomerName	String	客户姓名 示例值：xingming
Results	Array of InsuranceResult	结果
ReviewTime	String	复核时间 示例值：1970-01-01 00:00:00
ManualDetail	Array of UnderwriteConclusion	人工复核结果

错误码

最近更新时间：2021-03-10 08:13:06

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。
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。
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。
ServiceUnavailable	当前服务暂时不可用。
UnauthorizedOperation	未授权操作。
UnknownParameter	未知参数错误，用户多传未定义的参数会导致错误。
UnsupportedOperation	操作不支持。
UnsupportedProtocol	http(s) 请求协议错误，只支持 GET 和 POST 请求。
UnsupportedRegion	接口不支持所传地域。

业务错误码

错误码	说明
AuthFailure	CAM签名/鉴权错误。