人脸核身

API 文档

腾讯云

文档目录

API 文档
更新历史

最近更新时间：2025-06-11 01:30:32

第 84 次发布

发布时间：2025-06-11 01:30:22

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- GetDetectInfoEnhanced
  - 新增入参：IsReturnAllVideo

新增数据结构：

- VideoDetailData

修改数据结构：

- DetectInfoVideoData
  - 新增成员：LivenessVideos

第 83 次发布

发布时间：2025-05-26 19:47:42

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- GetFaceIdResult
  - 新增入参：IsEncryptResponse, Encryption
  - 新增出参：Encryption, EncryptedBody

第 82 次发布

发布时间：2024-12-17 01:30:37

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- ImageRecognitionV2
  - 新增入参：Extra
  - 新增出参：Extra

第 81 次发布

发布时间：2024-10-16 01:15:00

本次发布包含了以下内容：

改善已有的文档。

修改数据结构：

- DetectInfoText
  - 新增成员：LivenessInfoTag

第 80 次发布

发布时间：2024-09-25 01:20:56

本次发布包含了以下内容：

改善已有的文档。

修改数据结构：

- GetEidTokenConfig
  - 新增成员：Speed

第 79 次发布

发布时间：2024-09-24 01:19:58

本次发布包含了以下内容：

改善已有的文档。

修改数据结构：

- RuleIdConfig
  - 新增成员：Speed

第 78 次发布

发布时间：2024-08-21 01:40:11

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- CheckIdCardInformation
  - 新增入参：IsEncryptResponse, Encryption
  - 新增出参：EncryptedBody

第 77 次发布

发布时间：2024-08-19 01:41:40

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- DetectAIFakeFaces
  - 新增入参：Encryption, EncryptedBody
  - **修改入参：** FaceInput, FaceInputType

第 76 次发布

发布时间：2024-04-26 01:16:30

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- GetFaceIdResult
  - 新增出参：LivenessInfoTag, DeviceInfoLevel

第 75 次发布

发布时间：2024-04-22 01:17:05

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- PhoneVerification
  - 新增入参：VerifyMode
  - 新增出参：ResultDetail

第 74 次发布

发布时间：2024-04-12 01:15:49

本次发布包含了以下内容：

改善已有的文档。

修改数据结构：

- RuleIdConfig
  - 新增成员：MouthOpenRecognition

第 73 次发布

发布时间：2024-03-27 01:17:36

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- GetFaceIdToken
  - 新增入参：RuleId

第 72 次发布

发布时间：2024-03-26 01:17:00

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- GetFaceIdRiskInfo
- GetFaceidRiskInfoToken

第 71 次发布

发布时间：2024-03-25 01:16:50

本次发布包含了以下内容：

改善已有的文档。

修改数据结构：

- GetEidTokenConfig
  - 新增成员：MouthOpenRecognition

第 70 次发布

发布时间：2024-01-12 01:15:37

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- GetEidResult
  - 新增入参：IsCutIdCardImage, IsNeedIdCardAvatar

第 69 次发布

发布时间：2023-12-27 01:19:47

本次发布包含了以下内容：

改善已有的文档。

修改数据结构：

- IntentionQuestionResult
  - **修改成员：** FinalResultDetailCode, FinalResultMessage

第 68 次发布

发布时间：2023-12-22 01:16:17

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- DetectAIFakeFaces
  - 新增出参：ExtraInfo

新增数据结构：

- ExtraInfo
- RetrievalLivenessExtraInfo

第 67 次发布

发布时间：2023-12-06 01:49:35

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- ImageRecognitionV2

第 66 次发布

发布时间：2023-11-14 01:16:39

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- DetectAIFakeFaces

新增数据结构：

- AttackRiskDetail

第 65 次发布

发布时间：2023-11-13 11:06:52

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- GetDetectInfoEnhanced
  - 新增入参：IsEncryptResponse
  - 新增出参：EncryptedBody

第 64 次发布

发布时间：2023-11-10 01:16:33

本次发布包含了以下内容：

改善已有的文档。

修改数据结构：

- DetectInfoText
  - **修改成员：** IdInfoFrom, NFCRequestIds, NFCBillingCounts, PassNo, VisaNum

第 63 次发布

发布时间：2023-09-06 01:49:13

本次发布包含了以下内容：

改善已有的文档。

修改数据结构：

- ChargeDetail
  - 新增成员：IdCard
  - **废弃成员：** Idcard

第 62 次发布

发布时间：2023-08-31 01:16:46

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- GetEidResult
  - 新增出参：IntentionActionResult

修改数据结构：

- GetEidTokenConfig
  - 新增成员：IntentionActions

第 61 次发布

发布时间：2023-08-29 01:15:31

本次发布包含了以下内容：

改善已有的文档。

修改数据结构：

- IntentionVerifyData
  - **废弃成员：** AsrResultSimilarity

第 60 次发布

发布时间：2023-08-17 03:11:54

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- DetectAuth
  - 新增入参：IntentionActions
- GetDetectInfoEnhanced
  - 新增出参：IntentionActionResult

新增数据结构：

- IntentionActionConfig
- IntentionActionResult
- IntentionActionResultDetail

修改数据结构：

- RuleIdConfig
  - 新增成员：IntentionType

第 59 次发布

发布时间：2023-05-29 14:45:51

本次发布包含了以下内容：

改善已有的文档。

修改数据结构：

- DetectInfoText
  - 新增成员：UseIDType

第 58 次发布

发布时间：2023-05-23 01:21:54

本次发布包含了以下内容：

改善已有的文档。

修改数据结构：

- GetEidTokenConfig
  - 新增成员：IsSupportHMTResidentPermitOCR

第 57 次发布

发布时间：2023-04-17 01:20:00

本次发布包含了以下内容：

改善已有的文档。

修改数据结构：

- IntentionQuestionResult
  - 新增成员：FinalResultDetailCode, FinalResultMessage

第 56 次发布

发布时间：2023-04-13 01:17:40

本次发布包含了以下内容：

改善已有的文档。

修改数据结构：

- DetectInfoText

- 新增成员：PassNo, VisaNum

第 55 次发布

发布时间：2023-03-15 01:37:54

本次发布包含了以下内容：

改善已有的文档。

修改数据结构：

- DetectInfoText
  - 新增成员：NFCRequestIds, NFCBillingCounts

第 54 次发布

发布时间：2023-03-01 01:27:36

本次发布包含了以下内容：

改善已有的文档。

修改数据结构：

- DetectInfoText
  - 新增成员：IdInfoFrom

第 53 次发布

发布时间：2023-01-06 01:23:24

本次发布包含了以下内容：

改善已有的文档。

删除接口：

- GetRealNameAuthResult

第 52 次发布

发布时间：2023-01-05 01:23:16

本次发布包含了以下内容：

改善已有的文档。

删除接口：

- GetRealNameAuthToken

第 51 次发布

发布时间：2022-11-29 06:46:33

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- DetectAuth
  - 新增入参：Config

新增数据结构：

- RuleIdConfig

第 50 次发布

发布时间：2022-11-16 06:23:45

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- EncryptedPhoneVerification
  - 新增出参：ISP
- GetFaceIdToken
  - 新增入参：Encryption

第 49 次发布

发布时间：2022-11-11 06:30:28

本次发布包含了以下内容：

改善已有的文档。

删除接口：

- ApplyLivenessToken
- ApplySdkVerificationToken
- ApplyWebVerificationToken
- CreateUploadUrl
- DetectReflectLivenessAndCompare
- GenerateReflectSequence
- GetLivenessResult
- GetSdkVerificationResult
- GetWebVerificationResult
- VideoLivenessCompare

删除数据结构：

- CardVerifyResult
- CompareResult
- FileInfo
- VerificationDetail

第 48 次发布

发布时间：2022-10-17 06:26:28

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- ApplyLivenessToken
- ApplySdkVerificationToken
- ApplyWebVerificationToken
- CreateUploadUrl
- DetectReflectLivenessAndCompare
- GenerateReflectSequence
- GetLivenessResult
- GetSdkVerificationResult
- GetWebVerificationResult
- VideoLivenessCompare

新增数据结构：

- CardVerifyResult
- CompareResult

- FileInfo
- VerificationDetail

第 47 次发布

发布时间：2022-09-20 06:29:06

本次发布包含了以下内容：

改善已有的文档。

修改数据结构：

- DetectDetail
  - 新增成员：LivenessMode
- DetectInfoText
  - 新增成员：LivenessMode

第 46 次发布

发布时间：2022-09-02 06:31:36

本次发布包含了以下内容：

改善已有的文档。

修改数据结构：

- GetEidTokenConfig
  - 新增成员：IntentionRecognition

第 45 次发布

发布时间：2022-08-26 06:28:34

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- GetDetectInfoEnhanced
  - 新增入参：Encryption

修改数据结构：

- Encryption
  - 新增成员：Algorithm, TagList

第 44 次发布

发布时间：2022-08-03 18:41:29

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- GetEidResult
  - 新增出参：IntentionQuestionResult

修改数据结构：

- GetEidTokenConfig
  - 新增成员：IntentionMode, IntentionQuestions

第 43 次发布

发布时间：2022-07-30 15:12:30

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- DetectAuth
  - 新增入参：IntentionQuestions
- GetDetectInfoEnhanced
  - 新增出参：IntentionQuestionResult

新增数据结构：

- IntentionQuestion
- IntentionQuestionResult

第 42 次发布

发布时间：2022-07-28 06:11:52

本次发布包含了以下内容：

改善已有的文档。

修改数据结构：

- DetectInfoIdCardData
  - 新增成员：BackWarnInfos

第 41 次发布

发布时间：2022-07-14 17:47:55

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- GetEidResult
  - 新增出参：IntentionVerifyData

修改数据结构：

- GetEidTokenConfig
  - 新增成员：UseIntentionVerify, IntentionVerifyText

第 40 次发布

发布时间：2022-06-02 06:08:18

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- ParseNfcData

第 39 次发布

发布时间：2022-04-01 06:09:09

本次发布包含了以下内容：

改善已有的文档。

修改数据结构：

- IntentionVerifyData
  - 新增成员：AsrResultSimilarity

第 38 次发布

发布时间：2022-01-21 08:11:28

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- PhoneVerificationCMCC
- PhoneVerificationCTCC
- PhoneVerificationCUCC

第 37 次发布

发布时间：2022-01-07 08:08:34

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- GetWeChatBillDetails

新增数据结构：

- ChargeDetail
- WeChatBillDetail

第 36 次发布

发布时间：2022-01-06 08:09:38

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- PhoneVerification
  - 新增出参：Isp

修改数据结构：

- DetectInfoIdCardData
  - 新增成员：WarnInfos

第 35 次发布

发布时间：2021-12-16 08:09:22

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- DetectReflectLivenessAndCompare

第 34 次发布

发布时间：2021-12-10 08:06:32

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- DetectAuth
  - 新增入参：IntentionVerifyText
- GetDetectInfoEnhanced
  - 新增出参：IntentionVerifyData

新增数据结构：

- IntentionVerifyData

第 33 次发布

发布时间：2021-11-23 08:07:49

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- LivenessCompare
  - 新增入参：ImageUrl, VideoUrl
  - **修改入参：** ImageBase64, VideoBase64
- LivenessRecognition
  - 新增入参：VideoUrl
  - **修改入参：** VideoBase64

第 32 次发布

发布时间：2021-10-29 08:05:31

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- GetFaceIdToken
  - 新增入参：UseCos

修改数据结构：

- EidInfo
  - 新增成员：DesKey, UserInfo

第 31 次发布

发布时间：2021-10-21 11:26:59

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- GetEidToken
  - 新增入参：Encryption

第 30 次发布

发布时间：2021-08-24 08:06:24

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- GetFaceIdResult
  - 新增出参：DeviceInfoTag, RiskInfoTag

第 29 次发布

发布时间：2021-08-11 08:05:48

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- CheckIdNameDate

第 28 次发布

发布时间：2021-07-13 08:07:56

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- BankCard2EVerification
  - 新增入参：Encryption
- CheckBankCardInformation
  - 新增入参：Encryption
- CheckIdCardInformation
  - 新增入参：IsEncrypt
  - 新增出参：Encryption
- CheckPhoneAndName
  - 新增入参：Encryption
- IdCardVerification
  - 新增入参：Encryption
- ImageRecognition
  - 新增入参：Encryption
- LivenessRecognition
  - 新增入参：Encryption

第 27 次发布

发布时间：2021-07-07 10:00:09

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- CheckEidTokenStatus

修改接口：

- GetEidToken
  - 新增入参：RedirectUrl
  - 新增出参：Url

第 26 次发布

发布时间：2021-04-25 10:14:38

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- EncryptedPhoneVerification

第 25 次发布

发布时间：2021-04-01 08:04:52

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- GetDetectInfoEnhanced
  - 新增入参：IsEncrypt
  - 新增出参：Encryption
- IdCardOCRVerification
  - 新增入参：Encryption

第 24 次发布

发布时间：2021-03-25 08:04:55

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- GetEidResult
- GetEidToken

新增数据结构：

- EidInfo
- GetEidTokenConfig

第 23 次发布

发布时间：2021-03-24 08:04:57

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- MobileNetworkTimeVerification
  - 新增入参：Encryption
- MobileStatus
  - 新增入参：Encryption

第 22 次发布

发布时间：2021-03-16 08:05:37

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- BankCard4EVerification
  - 新增入参：Encryption
- BankCardVerification
  - 新增入参：Encryption

- DetectAuth
  - 新增入参：Encryption

第 21 次发布

发布时间：2021-03-03 08:04:35

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- MinorsVerification
  - 新增入参：Encryption

新增数据结构：

- Encryption

第 20 次发布

发布时间：2020-12-24 08:04:18

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- GetRealNameAuthResult
- GetRealNameAuthToken

第 19 次发布

发布时间：2020-11-27 08:03:44

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- CheckPhoneAndName

修改数据结构：

- DetectDetail
  - 新增成员：CompareLibType
- DetectInfoText
  - 新增成员：CompareLibType

第 18 次发布

发布时间：2020-10-16 08:04:13

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- GetFaceIdResult
- GetFaceIdToken

第 17 次发布

发布时间：2020-09-29 08:04:10

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- PhoneVerification
  - 新增入参：Iv

第 16 次发布

发布时间：2020-08-27 08:03:28

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- CheckBankCardInformation

第 15 次发布

发布时间：2020-08-18 08:12:45

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- CheckIdCardInformation
  - 新增出参：Quality

第 14 次发布

发布时间：2020-08-10 08:11:30

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- CheckIdCardInformation

第 13 次发布

发布时间：2020-08-07 08:11:16

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- PhoneVerification
  - 新增入参：CiphertextBlob, EncryptList

第 12 次发布

发布时间：2020-06-02 08:10:02

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- GetActionSequence
  - 新增入参：ActionType

- Liveness
  - 新增出参：BestFrameList
- LivenessCompare
  - 新增出参：BestFrameList
- LivenessRecognition
  - 新增出参：BestFrameList

第 11 次发布

发布时间：2020-03-05 22:50:23

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- GetDetectInfoEnhanced

新增数据结构：

- DetectDetail
- DetectInfoBestFrame
- DetectInfoIdCardData
- DetectInfoText
- DetectInfoVideoData

第 10 次发布

发布时间：2019-12-30 13:35:35

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- MobileNetworkTimeVerification
- MobileStatus

第 9 次发布

发布时间：2019-11-28 20:30:09

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- MinorsVerification
- PhoneVerification

第 8 次发布

发布时间：2019-09-12 17:08:04

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- IdCardOCRVerification

第 7 次发布

发布时间：2019-06-27 16:10:47

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- BankCard2EVerification
- BankCard4EVerification

第 6 次发布

发布时间：2019-06-06 21:54:47

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- Liveness

修改接口：

- BankCardVerification
  - 新增入参：CertType

第 5 次发布

发布时间：2019-03-21 19:55:03

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- BankCardVerification

第 4 次发布

发布时间：2019-02-22 15:15:07

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- IdCardVerification

第 3 次发布

发布时间：2019-01-17 19:49:49

本次发布包含了以下内容：

改善已有的文档。

修改接口：

- ImageRecognition
  - 新增出参：Result, Description
- LivenessCompare
  - 新增出参：Result, Description
- LivenessRecognition
  - 新增出参：Result, Description

第 2 次发布

发布时间：2018-12-20 19:44:02

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- GetActionSequence
- GetLiveCode
- ImageRecognition
- LivenessCompare
- LivenessRecognition

第 1 次发布

发布时间：2018-12-07 16:21:28

本次发布包含了以下内容：

改善已有的文档。

新增接口：

- DetectAuth
- GetDetectInfo

简介

最近更新时间：2025-05-26 19:47:49

产品概述

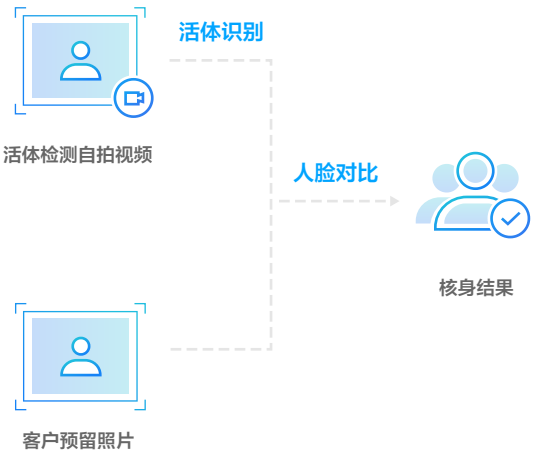
腾讯云慧眼人脸核身（原金融级身份认证身份）方案，是依托于证件OCR识别、活体检测、1：1人脸比对等AI技术，实现自然人真实身份核验的产品。秒级确认用户身份，帮助提升业务办理效率，降低人力成本。

功能介绍

人脸核身为您提供了OCR、活体检测、1:1人脸比对的配套服务，可根据您的需求灵活组合。

1.与权威数据库比对

2.与上传照片比对



身份证 OCR识别

支持识别身份证正反面，一次扫描即可返回身份证号、姓名、有效时间等所有字段，数字识别准确率达到 99.9% 以上，在多个字段上处于领先水平。处理倾斜、暗光、曝光、阴影等异常情况稳定性好，自适应判别纠正技术大大提高识别准确率。

活体检测

活体检测， 通过对一段实时录制的自拍视频进行检测，从而确认当前用户为真人，主要针对高安全性要求的人脸核身验证而研发的一种技术， 防止照片、视频、静态3D建模等各种不同类型的攻击； 目前支持唇语读数、动作、静默等多种活体模式，适用于多类验证场景。

人脸比对

人脸比对，根据面部特征，计算两张人脸的相似度，自动进行身份鉴别。立足于腾讯社交数据大平台收集的海量人脸训练集，结合高维 LBP、PCA、LDA 联合贝叶斯、度量学习、迁移学习、深度神经网络，2017年LFW 测评准确度为 99.80%。并支持多场景下的识别验证，对年龄、姿态及光强均有较好的支持度。

API 概览

最近更新时间：2024-10-17 01:19:46

其他接口

接口名称	接口功能	频率限制（次/秒）
GetWeChatBillDetails	查询账单明细（微信渠道）	20
ParseNfcData	获取证件NFC结果	20
GetFaceIdRiskInfo	获取SDK设备风险信息	20
GetFaceidRiskInfoToken	获取风险信息Token	20

人脸核身SaaS服务相关接口

接口名称	接口功能	频率限制（次/秒）
CheckEidTokenStatus	获取E证通Token状态	2000
DetectAuth	实名核身鉴权	100
GetDetectInfo	获取实名核身结果信息	50
GetDetectInfoEnhanced	获取实名核身结果信息增强版	100
GetEidResult	获取E证通结果信息	100
GetEidToken	获取E证通Token	100
GetFaceIdResult	获取SDK核验结果	100
GetFaceIdToken	获取SDKToken	100

实名信息核验相关接口

接口名称	接口功能	频率限制（次/秒）
IdCardVerification	身份信息认证（二要素核验）	100
IdCardOCRVerification	身份证识别及信息核验	100
CheckIdNameDate	身份信息及有效期核验	20
CheckBankCardInformation	银行卡基础信息查询	20
BankCard2EVerification	银行卡二要素核验	20
BankCardVerification	银行卡三要素核验	20
BankCard4EVerification	银行卡四要素核验	20
MobileNetworkTimeVerification	手机号在网时长核验	20
MobileStatus	手机号状态查询	20
PhoneVerification	手机号三要素核验	20
PhoneVerificationCMCC	手机号三要素核验（移动）	20
PhoneVerificationCTCC	手机号三要素核验（电信）	20
PhoneVerificationCUCC	手机号三要素核验（联通）	20
CheckPhoneAndName	手机号二要素核验	20
MinorsVerification	手机号实名查询	20

接口名称	接口功能	频率限制（次/秒）
EncryptedPhoneVerification	运营商三要素核验（加密）	20

人脸核身PaaS服务相关接口

接口名称	接口功能	频率限制（次/秒）
GetActionSequence	获取动作顺序	100
GetLiveCode	获取数字验证码	100
ImageRecognition	照片人脸核身	100
Liveness	活体检测	20
LivenessCompare	活体人脸比对	100
LivenessRecognition	活体人脸核身	100
ImageRecognitionV2	照片人脸核身(V2.0)	20
CheckIdCardInformation	身份证人像照片验真	100

AI人脸防护盾相关接口

接口名称	接口功能	频率限制（次/秒）
DetectAIFakeFaces	AI人脸防护盾	5

⚠ 注意：
以上给出的接口频率限制维度为 API + 接入地域 + 子账号，有关限频更多说明参考：API 频率限制说明

调用方式
请求结构

最近更新时间：2025-05-26 19:47:49

1. 服务地址

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

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

注意：对时延敏感的业务，建议指定带地域的域名。

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

目前支持的域名列表为：

接入地域	域名
就近地域接入（推荐，只支持非金融区）	faceid.tencentcloudapi.com
华南地区（广州）	faceid.ap-guangzhou.tencentcloudapi.com
华东地区（上海）	faceid.ap-shanghai.tencentcloudapi.com
华东地区（南京）	faceid.ap-nanjing.tencentcloudapi.com
华北地区（北京）	faceid.ap-beijing.tencentcloudapi.com
西南地区（成都）	faceid.ap-chengdu.tencentcloudapi.com
西南地区（重庆）	faceid.ap-chongqing.tencentcloudapi.com
港澳台地区（中国香港）	faceid.ap-hongkong.tencentcloudapi.com
亚太东南（新加坡）	faceid.ap-singapore.tencentcloudapi.com
亚太东南（雅加达）	faceid.ap-jakarta.tencentcloudapi.com
亚太东南（曼谷）	faceid.ap-bangkok.tencentcloudapi.com
亚太东北（首尔）	faceid.ap-seoul.tencentcloudapi.com
亚太东北（东京）	faceid.ap-tokyo.tencentcloudapi.com
美国东部（弗吉尼亚）	faceid.na-ashburn.tencentcloudapi.com
美国西部（硅谷）	faceid.na-siliconvalley.tencentcloudapi.com
南美地区（圣保罗）	faceid.sa-saopaulo.tencentcloudapi.com
欧洲地区（法兰克福）	faceid.eu-frankfurt.tencentcloudapi.com

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

金融区接入地域	金融区域名
华东地区（上海金融）	faceid.ap-shanghai-fsi.tencentcloudapi.com
华南地区（深圳金融）	faceid.ap-shenzhen-fsi.tencentcloudapi.com

2. 通信协议

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

3. 请求方法

支持的 HTTP 请求方法:

- POST（推荐）
- GET

POST 请求支持的 Content-Type 类型:

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

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

4. 字符编码

均使用 `UTF-8` 编码。

公共参数

最近更新时间：2025-01-15 01:17:35

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=AKID***/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，AKID*** 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 faceid；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********************************/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********************************/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********************************/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。

参数名称	类型	必选	描述
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*******************************

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*****************************
```

地域列表

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

地域	取值
华南地区（广州）	ap-guangzhou

签名方法 v3

最近更新时间：2024-12-25 01:36:37

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：AKID******************************** 和 ********************************。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=AKID********************************/2019-02-25/cvm/tc3_request, SignedHeaders=content-type;host;x-tc-action, Signature=10b1a37a7301a02ca19a647ad722d5e43b4b3cff309d421d85b46093f6ab6c4f" \
-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。
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

字段名称	解释
	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。
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 = "********************************"
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：
da98fb70dcf6b112dc21038d1eeeb3a95c74b4dcb12c1131f864f6066bd02be0，
8d70cbefb03939f929db64d32dc2ba89b1095620119fe3e050e2b18c5bd2752f，
b596b923aad85185e2d1f6659d2a062e0a86731226e021e61bfe06f7ed05f5af。

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 `********************************`。
Date	即 Credential 中的 Date 字段信息。此示例取值为 `2019-02-25`。
Service	即 Credential 中的 Service 字段信息。此示例取值为 `cvm`。

2）计算签名，伪代码如下：

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

此示例计算结果是 `10b1a37a7301a02ca19a647ad722d5e43b4b3cff309d421d85b46093f6ab6c4f`。

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 `AKID********************************`。
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 `10b1a37a7301a02ca19a647ad722d5e43b4b3cff309d421d85b46093f6ab6c4f`。

根据以上规则，示例中得到的值为：

```
TC3-HMAC-SHA256 Credential=AKID********************************/2019-02-25/cvm/tc3_request, SignedHeaders=content-type;host;x-tc-action, Signature=10b1a37a7301a02ca19a647ad722d5e43b4b3cff309d421d85b46093f6ab6c4f
```

最终完整的调用信息如下：

```
POST https://cvm.tencentcloudapi.com/
Authorization: TC3-HMAC-SHA256 Credential=AKID********************************/2019-02-25/cvm/tc3_request, SignedHeaders=content-type;host;x-tc-action, Signature=10b1a37a7301a02ca19a647ad722d5e43b4b3cff309d421d85b46093f6ab6c4f
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 AKID********************************
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 ********************************
```

```
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 AKID********************************
secret_id = os.environ.get("TENCENTCLOUD_SECRET_ID")
# 需要设置环境变量 TENCENTCLOUD_SECRET_KEY，值为示例的 ********************************
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 AKID********************************
secretId := os.Getenv("TENCENTCLOUD_SECRET_ID")
// 需要设置环境变量 TENCENTCLOUD_SECRET_KEY，值为示例的 ********************************
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 AKID********************************
$secretId = getenv("TENCENTCLOUD_SECRET_ID");
// 需要设置环境变量 TENCENTCLOUD_SECRET_KEY，值为示例的 ********************************
$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'
```

```ruby
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 AKID********************************
secret_id = ENV["TENCENTCLOUD_SECRET_ID"]
# 需要设置环境变量 TENCENTCLOUD_SECRET_KEY，值为示例的 ********************************
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=#{si
gnature}"
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 AKID********************************
string SECRET_ID = Environment.GetEnvironmentVariable("TENCENTCLOUD_SECRET_ID");
// 需要设置环境变量 TENCENTCLOUD_SECRET_KEY，值为示例的 ********************************
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 AKID********************************
const SECRET_ID = process.env.TENCENTCLOUD_SECRET_ID
// 需要设置环境变量 TENCENTCLOUD_SECRET_KEY，值为示例的 ********************************
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 AKID********************************
string SECRET_ID = getenv("TENCENTCLOUD_SECRET_ID");
// 需要设置环境变量 TENCENTCLOUD_SECRET_KEY，值为示例的 ********************************
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(const char* key, int key_len,
const char* input, int input_len,
unsigned char* output, unsigned int* output_len)
{
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, key_len, EVP_sha256(), NULL);
HMAC_Update(h, ( unsigned char* )input, input_len);
HMAC_Final(h, output, output_len);


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

}

void hex_encode(const char* input, int input_len, char* output)
{
static const char* const lut = "0123456789abcdef";

char add_out[128] = {0};
char temp[2] = {0};
for (size_t i = 0; i < input_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 AKID********************************
const char* SECRET_ID = getenv("TENCENTCLOUD_SECRET_ID");
// 需要设置环境变量 TENCENTCLOUD_SECRET_KEY，值为示例的 ********************************
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);
unsigned char k_date[64] = {0};
unsigned int output_len = 0;
hmac_sha256(k_key, strlen(k_key), date, strlen(date), k_date, &output_len);
unsigned char k_service[64] = {0};
hmac_sha256(k_date, output_len, service, strlen(service), k_service, &output_len);
unsigned char k_signing[64] = {0};
hmac_sha256(k_service, output_len, "tc3_request", strlen("tc3_request"), k_signing, &output_len);
unsigned char k_hmac_sha_sign[64] = {0};
hmac_sha256(k_signing, output_len, string_to_sign, strlen(string_to_sign), k_hmac_sha_sign, &output_len);

char signature[128] = {0};
```

```
hex_encode(k_hmac_sha_sign, output_len, 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。
AuthFailure.TokenFailure	临时证书 Token 错误。
AuthFailure.InvalidSecretId	密钥非法（不是云 API 密钥类型）。

签名方法

最近更新时间：2024-12-25 01:36:38

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

首次接触，建议使用 API Explorer 中的“签名串生成”功能，选择签名版本为“API 3.0 签名 v1”，可以生成签名过程进行验证，并提供了部分编程语言的签名示例，也可直接生成 SDK 代码。推荐使用腾讯云 API 配套的 8 种常见的编程语言 SDK，已经封装了签名和请求过程，均已开源，支持 Python、Java、PHP、Go、NodeJS、.NET、C++、Ruby。

推荐使用 API Explorer　　点击调试

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

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

1. 申请安全凭证

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

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

申请安全凭证的具体步骤如下：

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

注意：每个账号最多可以拥有两对 SecretId/SecretKey。

2. 生成签名串

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

假设用户的 SecretId 和 SecretKey 分别是：

- SecretId: AKID********************************
- SecretKey: ********************************

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

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

参数名称	中文	参数值
Action	方法名	DescribeInstances
SecretId	密钥 ID	AKID********************************
Timestamp	当前时间戳	1465185768
Nonce	随机正整数	11886
Region	实例所在区域	ap-guangzhou
InstanceIds.0	待查询的实例 ID	ins-09dx96dg
Offset	偏移量	0
Limit	最大允许输出	20
Version	接口版本号	2017-03-12

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

2.1. 对参数排序

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

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

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

2.2. 拼接请求字符串

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

然后将格式化后的各个参数用"&"拼接在一起，最终生成的请求字符串为:

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

2.3. 拼接签名原文字符串

此步骤生成签名原文字符串。
签名原文字符串由以下几个参数构成:

1. 请求方法: 支持 POST 和 GET 方式，这里使用 GET 请求，注意方法为全大写。
2. 请求主机:查看实例列表(DescribeInstances)的请求域名为：cvm.tencentcloudapi.com。实际的请求域名根据接口所属模块的不同而不同，详见各接口说明。
3. 请求路径: 当前版本云API的请求路径固定为 / 。
4. 请求字符串: 即上一步生成的请求字符串。

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

示例的拼接结果为:

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

2.4. 生成签名串

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

具体代码如下，以 PHP 语言为例:

```
$secretKey = '********************************';
$srcStr = 'GETcvm.tencentcloudapi.com/?Action=DescribeInstances&InstanceIds.0=ins-09dx96dg&Limit=20&Nonce=11886&Offset=0&R
egion=ap-guangzhou&SecretId=AKID********************************&Timestamp=1465185768&Version=2017-03-12';
```

```
$signStr = base64_encode(hash_hmac('sha1', $srcStr, $secretKey, true));
echo $signStr;
```

最终得到的签名串为：

```
7RAM2xfNMO9EiVTNmPg06MRnCvQ=
```

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

3. 签名串编码

生成的签名串并不能直接作为请求参数，需要对其进行 URL 编码。

如上一步生成的签名串为 7RAM2xfNMO9EiVTNmPg06MRnCvQ= ，最终得到的签名串请求参数（Signature）为：7RAM2xfNMO9EiVTNmPg06MRnCvQ%3D，它将用于生成最终的请求 URL。

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

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

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

4. 签名失败

根据实际情况，存在以下签名失败的错误码，请根据实际情况处理。

错误代码	错误描述
AuthFailure.SignatureExpire	签名过期
AuthFailure.SecretIdNotFound	密钥不存在
AuthFailure.SignatureFailure	签名错误
AuthFailure.TokenFailure	token 错误
AuthFailure.InvalidSecretId	密钥非法（不是云 API 密钥类型）

5. 签名演示

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

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

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

- Signature Demo

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

最终输出的 url 可能为：`https://cvm.tencentcloudapi.com/?Action=DescribeInstances&InstanceIds.0=ins-09dx96dg&Limit=20&Nonce=11886&Offset=0&Region=ap-`

guangzhou&SecretId=AKID********************************&Signature=7RAM2xfNMO9EiVTNmPg06MRnCvQ%3D&Timestamp=1465185768&Version=2017-03-12 。

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

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

注意：以下代码仅适用于 API 3.0，不能直接用于其他的签名流程，请以对应的实际文档为准。

Java

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

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

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

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

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

public static void main(String[] args) throws Exception {
TreeMap<String, Object> params = new TreeMap<String, Object>(); // TreeMap可以自动排序
// 实际调用时应当使用随机数，例如: params.put("Nonce", new Random().nextInt(java.lang.Integer.MAX_VALUE));
params.put("Nonce", 11886); // 公共参数
// 实际调用时应当使用系统当前时间，例如: params.put("Timestamp", System.currentTimeMillis() / 1000);
params.put("Timestamp", 1465185768); // 公共参数
// 需要设置环境变量 TENCENTCLOUD_SECRET_ID，值为示例的 AKID********************************
params.put("SecretId", System.getenv("TENCENTCLOUD_SECRET_ID")); // 公共参数
```

```
params.put("Action", "DescribeInstances"); // 公共参数
params.put("Version", "2017-03-12"); // 公共参数
params.put("Region", "ap-guangzhou"); // 公共参数
params.put("Limit", 20); // 业务参数
params.put("Offset", 0); // 业务参数
params.put("InstanceIds.0", "ins-09dx96dg"); // 业务参数
// 需要设置环境变量 TENCENTCLOUD_SECRET_KEY，值为示例的 ********************************
params.put("Signature", sign(getStringToSign(params), System.getenv("TENCENTCLOUD_SECRET_KEY"), "HmacSHA1")); // 公共参数
System.out.println(getUrl(params));
}
}
```

Python

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

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

import requests

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

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

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

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

```
# resp = requests.get("https://" + endpoint, params=data)
# print(resp.url)
```

Golang

```
package main

import (
"bytes"
"crypto/hmac"
"crypto/sha1"
"encoding/base64"
"fmt"
"os"
"sort"
"strconv"
)

func main() {
// 需要设置环境变量 TENCENTCLOUD_SECRET_ID，值为示例的 AKID********************************
secretId := os.Getenv("TENCENTCLOUD_SECRET_ID")
// 需要设置环境变量 TENCENTCLOUD_SECRET_KEY，值为示例的 ********************************
secretKey := os.Getenv("TENCENTCLOUD_SECRET_KEY")
params := map[string]string{
"Nonce": "11886",
"Timestamp": strconv.Itoa(1465185768),
"Region": "ap-guangzhou",
"SecretId": secretId,
"Version": "2017-03-12",
"Action": "DescribeInstances",
"InstanceIds.0": "ins-09dx96dg",
"Limit": strconv.Itoa(20),
"Offset": strconv.Itoa(0),
}

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

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

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

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

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

PHP

```
<?php
// 需要设置环境变量 TENCENTCLOUD_SECRET_ID，值为示例的 AKID********************************
$secretId = getenv("TENCENTCLOUD_SECRET_ID");
// 需要设置环境变量 TENCENTCLOUD_SECRET_KEY，值为示例的 ********************************
$secretKey = getenv("TENCENTCLOUD_SECRET_KEY");
$param["Nonce"] = 11886;//rand();
$param["Timestamp"] = 1465185768;//time();
$param["Region"] = "ap-guangzhou";
$param["SecretId"] = $secretId;
$param["Version"] = "2017-03-12";
$param["Action"] = "DescribeInstances";
$param["InstanceIds.0"] = "ins-09dx96dg";
$param["Limit"] = 20;
$param["Offset"] = 0;

ksort($param);

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

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

Ruby

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

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

```
secret_key = ENV["TENCENTCLOUD_SECRET_KEY"]

method = 'GET'
endpoint = 'cvm.tencentcloudapi.com'
data = {
'Action' => 'DescribeInstances',
'InstanceIds.0' => 'ins-09dx96dg',
'Limit' => 20,
'Nonce' => 11886,
'Offset' => 0,
'Region' => 'ap-guangzhou',
'SecretId' => secret_id,
'Timestamp' => 1465185768, # Time.now.to_i
'Version' => '2017-03-12',
}
sign = method + endpoint + '/?'
params = []
data.sort.each do |item|
params << "#{item[0]}=#{item[1]}"
end
sign += params.join('&')
digest = OpenSSL::Digest.new('sha1')
data['Signature'] = Base64.encode64(OpenSSL::HMAC.digest(digest, secret_key, sign))
puts data['Signature']

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

DotNet

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

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

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

public static void Main(string[] args)
{
// 密钥参数
// 需要设置环境变量 TENCENTCLOUD_SECRET_ID，值为示例的 AKID********************************
string SECRET_ID = Environment.GetEnvironmentVariable("TENCENTCLOUD_SECRET_ID");
// 需要设置环境变量 TENCENTCLOUD_SECRET_KEY，值为示例的 ********************************
string SECRET_KEY = Environment.GetEnvironmentVariable("TENCENTCLOUD_SECRET_KEY");

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

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

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

NodeJS

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

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

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

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

function main(){
// 密钥参数
// 需要设置环境变量 TENCENTCLOUD_SECRET_ID，值为示例的 AKID********************************
const SECRET_ID = process.env.TENCENTCLOUD_SECRET_ID
// 需要设置环境变量 TENCENTCLOUD_SECRET_KEY，值为示例的 ********************************
const SECRET_KEY = process.env.TENCENTCLOUD_SECRET_KEY

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

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

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

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

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

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

返回结果

最近更新时间：2024-03-12 19:51:28

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

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

正确返回结果

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

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

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

错误返回结果

若调用失败，其返回值示例如下为：

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

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

公共错误码

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

参数类型

最近更新时间：2022-08-10 06:31:53

目前腾讯云 API 3.0 输入参数和输出参数支持如下几种数据格式：

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

人脸核身SaaS服务相关接口
获取E证通Token状态

最近更新时间：2025-05-26 19:47:56

1. 接口描述

接口请求域名： faceid.tencentcloudapi.com 。

用于轮询E证通H5场景EidToken验证状态。

默认接口请求频率限制：2000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：CheckEidTokenStatus。
Version	是	String	公共参数，本接口取值：2018-03-01。
Region	否	String	公共参数，本接口不需要传递此参数。
EidToken	是	String	E证通流程的唯一标识，调用GetEidToken接口时生成。 示例值：2B3B265E-2C91-5A62-B32D-D0CA5C3F1A15

3. 输出参数

参数名称	类型	描述
Status	String	状态。 - init：EidToken未验证。 - doing: EidToken验证中。 - finished: EidToken验证完成。 - timeout: EidToken已超时。 示例值：init
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 获取E证通Token状态成功示例

输入示例

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

{
```

```
"EidToken": "2B3B265E-2C91-5A62-B32D-D0CA5C3F1A15"
}
```

输出示例

```
{
"Response": {
"RequestId": "dd4e965c-5a6f-43ee-7764-8272e16b85a3",
"Status": "doing"
}
}
```

示例2 获取E政通Token状态失败示例

输入示例

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

{
"EidToken": "3B3B265E-3C91-5A62-B32D-D0CA5C3F1A15"
}
```

输出示例

```
{
"Response": {
"Error": {
"Code": "InvalidParameterValue.BizTokenIllegal",
"Message": "BizToken不合法。"
},
"RequestId": "9ad38b82-7b5c-4bb1-b6c0-44ce86bc148e"
}
}
```

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.DbError	数据库异常。
FailedOperation.UnKnown	内部未知错误。
InternalError	内部错误。
InvalidParameter	参数错误。
InvalidParameterValue.BizTokenExpired	BizToken过期。
InvalidParameterValue.BizTokenIllegal	BizToken不合法。
UnauthorizedOperation.Arrears	账号已欠费。
UnauthorizedOperation.Nonactivated	未开通服务。

实名核身鉴权

最近更新时间：2025-05-26 19:47:56

1. 接口描述

接口请求域名： faceid.tencentcloudapi.com 。

每次调用人脸核身SaaS化服务前，需先调用本接口获取BizToken，用来串联核身流程，在验证完成后，用于获取验证结果信息。

默认接口请求频率限制：100次/秒。

推荐使用 API Explorer　点击调试

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

2. 输入参数

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

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：DetectAuth。
Version	是	String	公共参数，本接口取值：2018-03-01。
Region	否	String	公共参数，本接口不需要传递此参数。
RuleId	是	String	业务流程ID。 - 用于细分客户使用场景，可为业务配置不同的业务流程。 - 申请开通服务后，登录腾讯云慧眼人脸核身控制进行创建，审核通过后即可调用。 - 如有疑问，请添加腾讯云人脸核身小助手进行咨询。 示例值：1
TerminalType	否	String	本接口不需要传递此参数。
IdCard	否	String	验证人的身份证号码。 - 是否必传基于控制台申请业务流程时配置的提示。 示例值：11204416541220243X
Name	否	String	验证人的姓名。 - 是否必传基于控制台申请业务流程时配置的提示。 - 最长长度32位。中文请使用UTF-8编码。 示例值：韦小宝
RedirectUrl	否	String	认证结束后重定向的回调链接地址。 - 最长长度1024位。 示例值：https://www.qq.com
Extra	否	String	透传字段，在获取验证结果时返回。 - 最长长度1024位。 示例值：Orderid=f904f4cf75db4f8fdc4f942c7f7a
ImageBase64	否	String	用于人脸比对的图像数据，使用base64编码。 - Base64编码后的图片数据大小不超过3M。 - 仅支持jpg、png格式。 - 请使用标准的Base64编码方式(带=补位)，编码规范参考RFC4648。 示例值：/9j/4AAQSkZJRg.....s97n//2Q==
Encryption	否	Encryption	敏感数据加密信息。对传入信息（姓名、身份证号）有加密需求的用户可使用此参数，详情请点击左侧链接。
IntentionVerifyText	否	String	意愿核身（朗读模式）使用的文案。 - 若未使用意愿核身（朗读模式），则该字段无需传入。 - 最长可接受120的字符串长度。

参数名称	必选	类型	描述
IntentionQuestions.N	否	Array of IntentionQuestion	意愿核身（语音播报+语音回答模式）使用的文案。 - 包括：系统语音播报的文本、需要核验的标准文本。 - 问答模式支持1-10轮（不超过10轮）的意愿确认。
IntentionActions.N	否	Array of IntentionActionConfig	意愿核身（点头确认模式）使用的文案。 - 若未使用意愿核身（点头确认模式），则该字段无需传入。 - 点头确认模式支持1-10轮（不超过10轮）的意愿确认。
Config	否	RuleIdConfig	意愿核身流程配置。

3. 输出参数

参数名称	类型	描述
Url	String	用于发起核身流程的URL，仅微信H5场景使用。 示例值：https://faceid.qq.com?token=81EEF678-28EE-4759-A82E-6CBBBE6BC442
BizToken	String	一次核验流程的唯一标识。 - 有效时间为7,200秒，超过有效期再进行人脸核验会报错，请在有效期内进行核验。 - 完成人脸核验后，需根据此标识调用获取实名核身结果信息增强版获取用户最终验证结果信息。 示例值：81EEF6178-28EE-4759-A823E-6CBBBE6BC442
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 微信小程序获取BizToken成功示例

输入示例

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

{
"RuleId": "0",
"IdCard": "11204416541220243X",
"Name": "韦小宝"
}
```

输出示例

```
{
"Response": {
"BizToken": "CE661F1A-0F1E-45BD-BE13-34C05CEA7681",
"Url": "https://open.weixin.qq.com/connect/oauth2/authorize?appid=wx2cca36a86d5035ae&redirect_uri=http%3A%2F%2Fopen.faceid.qq.com%2Fv1%2Fapi%2FgetCode%3FbizRedirect%3Dhttp%253A%252F%252Ffaceid.qq.com%252Fapi%252Fauth%252FgetOpenidAndSaveToken%253Ftoken%253DCE661F1A-0F1E-45BD-BE13-34C05CEA7681&response_type=code&scope=snsapi_base&state=&component_appid=wx9802ee81e68d6dee#wechat_redirect",
"RequestId": "f904f4cf-75db-4f8f-a5ec-dc4f942c7f7a"
}
}
```

示例2 H5获取BizToken成功示例

输入示例

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

{
"RuleId": "0",
"IdCard": "11204416541220243X",
"Name": "韦小宝",
"Extra": "Orderid=f904f4cf75db4f8fdc4f942c7f7a",
"RedirectUrl": "http://www.qq.com"
}
```

输出示例

```
{
"Response": {
"BizToken": "CE661F1A-0F1E-45BD-BE13-34C05CEA7681",
"Url": "https://open.weixin.qq.com/connect/oauth2/authorize?appid=wx2cca36a86d5035ae&redirect_uri=http%3A%2F%2Fopen.faceid.qq.com%2Fv1%2Fapi%2FgetCode%3FbizRedirect%3Dhttp%253A%252F%252Ffaceid.qq.com%252Fapi%252Fauth%252FgetOpenidAndSaveToken%253Ftoken%253DCE661F1A-0F1E-45BD-BE13-34C05CEA7681&response_type=code&scope=snsapi_base&state=&component_appid=wx9802ee81e68d6dee#wechat_redirect",
"RequestId": "f904f4cf-75db-4f8f-a5ec-dc4f942c7f7a"
}
}
```

示例3 获取BizToken失败示例

输入示例

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

{
"RuleId": "999",
"IdCard": "11204416541220243X",
"Name": "韦小宝"
}
```

输出示例

```
{
"Response": {
"Error": {
"Code": "InvalidParameterValue.RuleIdNotExist",
"Message": "RuleId不存在，请到人脸核身控制台申请。"
},
"RequestId": "e90c9abf-dcb3-4efa-97fc-5c4501b8182c"
```

```
}
}
```

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.InvalidAuthorization	CAM签名/鉴权错误。
FailedOperation.DecryptSystemError	解密失败。
FailedOperation.UnKnown	内部未知错误。
InvalidParameter	参数错误。
InvalidParameter.UnsupportEncryptField	存在不加密的字段，请参考文档修改。
InvalidParameterValue.RuleIdDisabled	该ruleid已被您停用，请确认后重试。
InvalidParameterValue.RuleIdNotExist	RuleId不存在，请到人脸核身控制台申请。
UnauthorizedOperation.ActivateError	服务开通异常。
UnauthorizedOperation.Activating	服务开通中。
UnauthorizedOperation.Arrears	账号已欠费。
UnauthorizedOperation.NonAuthorize	账号未实名。
UnauthorizedOperation.Nonactivated	未开通服务。

获取实名核身结果信息增强版

最近更新时间：2025-06-11 01:30:30

1. 接口描述

接口请求域名： faceid.tencentcloudapi.com 。

完成验证后，用BizToken调用本接口获取结果信息，BizToken生成后三天内（3*24*3,600秒）可多次拉取。

默认接口请求频率限制：100次/秒。

推荐使用 API Explorer 点击调试

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

2. 输入参数

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

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：GetDetectInfoEnhanced。
Version	是	String	公共参数，本接口取值：2018-03-01。
Region	否	String	公共参数，本接口不需要传递此参数。
BizToken	是	String	人脸核身流程的标识，调用DetectAuth接口时生成。 示例值：CF742D11-BD49-4704-A33F-B9372D6322D6
RuleId	是	String	用于细分客户使用场景，由腾讯侧在线下对接时分配。 示例值：5
InfoType	否	String	指定拉取的结果信息。 - 取值（0：全部；1：文本类；2：身份证信息；3：视频最佳截图信息）。 - 例如 13 表示拉取文本类、视频最佳截图信息。 - 默认值：0 示例值：13
BestFramesCount	否	Integer	从活体视频中截取一定张数的最佳帧。 - 仅部分服务支持，若需使用请与慧眼小助手沟通。 - 默认值为0，最大值为10，超出10的最多只给10张。 - InfoType需要包含3。 示例值：1
IsCutIdCardImage	否	Boolean	是否对身份证照片进行裁边。 - 默认为false。 - InfoType需要包含2。 示例值：false
IsNeedIdCardAvatar	否	Boolean	是否需要从身份证中抠出头像。 - 默认为false。 - InfoType需要包含2。 示例值：false
IsEncrypt	否	Boolean	已弃用。 示例值：false
Encryption	否	Encryption	是否需要对返回中的敏感信息进行加密。 - 只需指定加密算法Algorithm即可，其余字段传入默认值。 - 敏感信息包括：Response.Text.IdCard、Response.Text.Name、Response.Text.OcrIdCard、Response.Text.OcrName。
IsEncryptResponse	否	Boolean	是否对回包整体进行加密。 示例值：false

参数名称	必选	类型	描述
IsReturnAllVideo	否	Boolean	是否需要返回认证中间过程的刷脸重试视频，默认不开启，多段视频需要存储到COS空间中，因此开启后还需要额外开启数据存储服务才可生效。详见数据存储指引。 示例值：false

3. 输出参数

参数名称	类型	描述
Text	DetectInfoText	文本类信息。 注意：此字段可能返回 null，表示取不到有效值。
IdCardData	DetectInfoIdCardData	身份证照片信息。 注意：此字段可能返回 null，表示取不到有效值。
BestFrame	DetectInfoBestFrame	最佳帧信息。 注意：此字段可能返回 null，表示取不到有效值。
VideoData	DetectInfoVideoData	视频信息。 注意：此字段可能返回 null，表示取不到有效值。
Encryption	Encryption	敏感数据加密信息。 注意：此字段可能返回 null，表示取不到有效值。
IntentionVerifyData	IntentionVerifyData	意愿核身朗读模式结果信息。 - 若未使用意愿核身功能，该字段返回值可以不处理。 注意：此字段可能返回 null，表示取不到有效值。
IntentionQuestionResult	IntentionQuestionResult	意愿核身问答模式结果。 - 若未使用该意愿核身功能，该字段返回值可以不处理。 注意：此字段可能返回 null，表示取不到有效值。
IntentionActionResult	IntentionActionResult	意愿核身点头确认模式的结果信息。 - 若未使用该意愿核身功能，该字段返回值可以不处理。 注意：此字段可能返回 null，表示取不到有效值。
EncryptedBody	String	加密后的数据。 注意：此字段可能返回 null，表示取不到有效值。 示例值：0hMvx6fX6n...gXE0adJA==
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 获取结果信息成功示例

输入示例

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

{
"InfoType": "1",
"BizToken": "CE661F1A-0F1E-45BD-BE13-34C05CEA7681",
"RuleId": "0"
}
```

输出示例

```
{
"Response": {
"BestFrame": {
"BestFrame": "/9j/4AAQSk...JKD2A//9k=",
"BestFrames": [
"/9j/4AAQSk...002dgP/9k=",
"/9j/4AAQSk...vx+YH/2Q==",
"/9j/4AAQSk...n6fj5Af//Z",
"/9j/4AAQSk...dzlFFagf/Z"
]
},
"EncryptedBody": "",
"Encryption": {
"Algorithm": "",
"CiphertextBlob": "",
"EncryptList": [],
"Iv": "",
"TagList": []
},
"IdCardData": {
"Avatar": null,
"BackWarnInfos": null,
"OcrBack": null,
"OcrFront": null,
"ProcessedBackImage": null,
"ProcessedFrontImage": null,
"WarnInfos": null
},
"IntentionActionResult": null,
"IntentionQuestionResult": {
"AsrResult": [],
"Audios": [],
"FinalResultCode": null,
"FinalResultDetailCode": null,
"FinalResultMessage": null,
"ResultCode": [],
"ScreenShot": [],
"Video": null
},
"IntentionVerifyData": {
"AsrResult": null,
"AsrResultSimilarity": null,
"ErrorCode": null,
"ErrorMessage": null,
"IntentionVerifyBestFrame": null,
"IntentionVerifyVideo": null
},
"RequestId": "91173d84-e461-4da4-a270-2b8d48a2e136",
"Text": {
"CompareLibType": "权威库",
"Comparemsg": "成功",
"Comparestatus": 0,
"ErrCode": 0,
"ErrMsg": "成功",
"Extra": "",
"IdCard": "11204416541220243X",
```

```
"IdInfoFrom": "其他",
"LiveMsg": "成功",
"LiveStatus": 0,
"LivenessDetail": [
{
"CompareLibType": "权威库",
"Comparemsg": "成功",
"Comparestatus": 0,
"Errcode": 0,
"Errmsg": "成功",
"Idcard": "11204416541220243X",
"IsNeedCharge": true,
"Livemsg": "成功",
"LivenessMode": 1,
"Livestatus": 0,
"Name": "韦小宝",
"ReqTime": "1730444265275",
"Seq": "1f330eea-f5db-4726-a7b4-38cdf1aefb02",
"Sim": "95.51"
}
],
"LivenessInfoTag": null,
"LivenessMode": 1,
"Location": null,
"Mobile": "",
"NFCBillingCounts": 0,
"NFCRequestIds": [],
"Name": "韦小宝",
"OcrAddress": null,
"OcrAuthority": null,
"OcrBirth": null,
"OcrGender": null,
"OcrIdCard": "",
"OcrName": "",
"OcrNation": null,
"OcrValidDate": null,
"PassNo": null,
"Sim": "95.51",
"UseIDType": 0,
"VisaNum": null
},
"VideoData": {
"LivenessVideo": "AAAAGGZ0eX...VyYWxidW0h"
}
}
}
```


示例2 获取结果信息失败示例

输入示例

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

```
{
"InfoType": "2",
"BizToken": "4237FAC9-1AE4-4954-B495-FB07D91141CA",
"RuleId": "0"
}
```

输出示例

```
{
"Response": {
"Error": {
"Code": "InvalidParameter",
"Message": "非法BizToken。"
},
"RequestId": "0832c071-0c3e-4a1f-b7d3-4910221a1e18"
}
}
```

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.DecryptSystemError	解密失败。
FailedOperation.EncryptSystemError	加密失败。
FailedOperation.StsUnAuthErrError	STS未授权。
FailedOperation.UnKnown	内部未知错误。
InvalidParameter	参数错误。
InvalidParameterValue.BizTokenExpired	BizToken过期。

错误码	描述
InvalidParameterValue.RuleIdNotExist	RuleId不存在，请到人脸核身控制台申请。
UnauthorizedOperation.ActivateError	服务开通异常。
UnauthorizedOperation.Activating	服务开通中。
UnauthorizedOperation.Arrears	账号已欠费。
UnauthorizedOperation.NonAuthorize	账号未实名。
UnauthorizedOperation.Nonactivated	未开通服务。

获取E证通结果信息

最近更新时间：2025-05-26 19:47:56

1. 接口描述

接口请求域名： faceid.tencentcloudapi.com 。

完成验证后，用EidToken调用本接口获取结果信息，EidToken生成后三天内（3*24*3,600秒）可多次拉取。

默认接口请求频率限制：100次/秒。

推荐使用 API Explorer　点击调试

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

2. 输入参数

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

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：GetEidResult。
Version	是	String	公共参数，本接口取值：2018-03-01。
Region	否	String	公共参数，本接口不需要传递此参数。
EidToken	是	String	E证通流程的唯一标识，调用GetEidToken接口时生成。 示例值：CE661F1A-0F1E-45BD-BE133-34C05CEA76812
InfoType	否	String	指定需要拉取的结果信息。 - 取值范围： 0：全部。 1：文本类。 2：身份证信息。 3：最佳截图信息。 5：意愿核身朗读模式相关结果。 6：意愿核身问答/点头模式相关结果。 - 例如 13表示拉取文本类、最佳截图信息。 - 默认值：0 示例值：1
BestFramesCount	否	Integer	从活体视频中截取一定张数的最佳帧。 - 默认为0，最大为3，超出3的最多只给3张。 - InfoType需要包含3。 示例值：1
IsCutIdCardImage	否	Boolean	是否对身份证照片进行裁边。 - 默认为false。 - InfoType需要包含2。 示例值：false
IsNeedIdCardAvatar	否	Boolean	是否需要从身份证中抠出头像。 - 默认为false。 - InfoType需要包含2。 示例值：false

3. 输出参数

参数名称	类型	描述
Text	DetectInfoText	文本类信息。 - 基于对敏感信息的保护，验证使用的姓名和身份证号统一通过加密后从EidInfo参数中返回。 - 如需获取请在控制台申请返回身份信息，详见E证通获取实名信息指引。 注意：此字段可能返回 null，表示取不到有效值。

参数名称	类型	描述
IdCardData	DetectInfoIdCardData	身份证照片信息。 注意：此字段可能返回 null，表示取不到有效值。
BestFrame	DetectInfoBestFrame	最佳帧信息。 注意：此字段可能返回 null，表示取不到有效值。
EidInfo	EidInfo	Eid信息。 - EidInfo字段只有在人脸核身控制台完成“申请返回实名信息”之后返回，操作指引详见 E证通获取实名信息指引。 - Eid信息包括商户下用户唯一标识以及加密后的姓名、身份证号信息。 - 解密方式详见 E证通获取实名信息指引 - 只有整个核验流程完成之后才能返回该字段信息。 注意：此字段可能返回 null，表示取不到有效值。
IntentionVerifyData	IntentionVerifyData	意愿核身朗读模式相关信息。 - 若未使用意愿核身朗读功能，该字段返回值可以不处理。 注意：此字段可能返回 null，表示取不到有效值。
IntentionQuestionResult	IntentionQuestionResult	意愿核身问答模式相关信息。 - 若未使用意愿核身问答模式功能，该字段返回值可以不处理。 注意：此字段可能返回 null，表示取不到有效值。
IntentionActionResult	IntentionActionResult	意愿核身点头确认模式的结果信息。 - 若未使用该意愿核身功能，该字段返回值可以不处理。 注意：此字段可能返回 null，表示取不到有效值。
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 获取E证通结果信息成功示例

成功获取E证通结果信息。

输入示例

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

{
"InfoType": "1",
"EidToken": "CE661F1A-0F1E-45BD-BE133-34C05CEA76812"
}
```

输出示例

```
{
"Response": {
"BestFrame": {
"BestFrame": "/9j/4AAQSk...UVoYn/2Q==",
"BestFrames": [
"/9j/4AAQSk...AT/U//2Q=="
]
},
"EidInfo": {
"DesKey": "",
"EidCode": "OMKhJ5hkbPnA5PKGrzD1oKZ/e8W9w2g1yAOUIry7G/4xMDAw",
```

```
"EidSign": "MEQCIDt0Xa...YygK4Fhw==",
"UserInfo": ""
},
"IdCardData": {
"Avatar": "",
"BackWarnInfos": null,
"OcrBack": "",
"OcrFront": "",
"ProcessedBackImage": "",
"ProcessedFrontImage": "",
"WarnInfos": null
},
"IntentionActionResult": null,
"IntentionQuestionResult": {
"AsrResult": null,
"Audios": null,
"FinalResultCode": "",
"FinalResultDetailCode": null,
"FinalResultMessage": null,
"ResultCode": null,
"ScreenShot": null,
"Video": ""
},
"IntentionVerifyData": {
"AsrResult": "",
"AsrResultSimilarity": "",
"ErrorCode": 0,
"ErrorMessage": "",
"IntentionVerifyBestFrame": "",
"IntentionVerifyVideo": ""
},
"RequestId": "29a30ec8-0c15-4566-a5ab-99f88a3f0821",
"Text": {
"CompareLibType": "权威库",
"Comparemsg": "成功",
"Comparestatus": 0,
"ErrCode": 0,
"ErrMsg": "成功",
"Extra": "",
"IdCard": "",
"IdInfoFrom": "其他",
"LiveMsg": "成功",
"LiveStatus": 0,
"LivenessInfoTag": [
"01"
],
"LivenessDetail": [
{
"CompareLibType": "权威库",
"Comparemsg": "成功",
"Comparestatus": 0,
"Errcode": 0,
"Errmsg": "成功",
"Idcard": "",
"IsNeedCharge": true,
"Livemsg": "成功",
"LivenessMode": 4,
```

```
"Livestatus": 0,
"Name": "",
"ReqTime": "1730451371368",
"Seq": "092ff8ca-4d1c-4c2122-b513-ec8ced564c9a",
"Sim": "97.31"
}
],
"LivenessMode": 4,
"Location": "",
"Mobile": "",
"NFCBillingCounts": 0,
"NFCRequestIds": null,
"Name": "",
"OcrAddress": "",
"OcrAuthority": "",
"OcrBirth": "",
"OcrGender": "",
"OcrIdCard": "",
"OcrName": "",
"OcrNation": "",
"OcrValidDate": "",
"PassNo": "",
"Sim": "97.31",
"UseIDType": 0,
"VisaNum": ""
}
}
}
```


示例2 获取E证通结果信息失败示例

获取E证通结果信息失败，传入过期BizToken。

输入示例

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

{
"InfoType": "1",
"EidToken": "CE661F1A-0F1E-45B1D-BE133-34C05CEA76812"
}
```

输出示例


```
{
"Response": {
"Error": {
"Code": "InvalidParameterValue.BizTokenExpired",
"Message": "BizToken过期。"
},
"RequestId": "19f36e49-d419-48a7-b86a-edcb81e71909"
```

```
}
}
```

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.DecryptSystemError	解密失败。
FailedOperation.StsUnAuthErrError	STS未授权。
FailedOperation.UnKnown	内部未知错误。
InternalError	内部错误。
InternalError.EncryptSystemError	加密失败。
InvalidParameter	参数错误。
InvalidParameterValue.BizTokenExpired	BizToken过期。
InvalidParameterValue.BizTokenIllegal	BizToken不合法。
UnauthorizedOperation.Arrears	账号已欠费。
UnauthorizedOperation.NonAuthorize	账号未实名。
UnauthorizedOperation.Nonactivated	未开通服务。

获取E证通Token

最近更新时间：2025-05-26 19:47:55

1. 接口描述

接口请求域名： faceid.tencentcloudapi.com 。

每次调用E证通服务前，需先调用本接口获取EidToken，用来串联E证通流程，在验证完成后，用于获取E证通结果信息。

默认接口请求频率限制：100次/秒。

推荐使用 API Explorer 点击调试

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

2. 输入参数

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

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：GetEidToken。
Version	是	String	公共参数，本接口取值：2018-03-01。
Region	否	String	公共参数，本接口不需要传递此参数。
MerchantId	是	String	EID商户ID。 - 商户ID通过人脸核身控制台自助接入申请。 - 商户ID与您通过腾讯云人脸核身控制台完成自助接入时所使用的腾讯云账号绑定。 - 必须使用申请该商户ID时登录的腾讯云账号所对应的腾讯云API密钥调用该接口。 示例值：0NSJ2432062616009321 23
IdCard	否	String	身份标识。 - 未使用OCR服务时，必须传入。 - 规则：a-z，A-Z，0-9组合。 - 最长长度32位。 示例值：112044165412202 43X
Name	否	String	姓名。 - 未使用OCR服务时，必须传入。 - 最长长度32位。 - 中文请使用UTF-8编码。 示例值：韦小宝
Extra	否	String	透传字段，在获取验证结果时返回。 - 最长长度1024位。 示例值：20241101170759199020006 0000
Config	否	GetEidTokenConfig	小程序模式配置，包括如何传入姓名身份证的配置，以及是否使用意愿核身。
RedirectUrl	否	String	用户从Url中进入核身认证结束后重定向的回调链接地址。 - 最长长度1024位。 - EidToken会在该链接的query参数中。 示例值：https://faceid.qq.com
Encryption	否	Encryption	敏感数据加密信息。 - 对传入信息（姓名、身份证号）有加密需求的用户可使用此参数，详情请点击左侧链接。

3. 输出参数

参数名称	类型	描述
EidToken	String	一次核身流程的标识。 - 有效时间为600秒。

参数名称	类型	描述
		- 完成核身后，可用该标识获取验证结果信息。 示例值：CE6621F1A-0F1E-45BD-BE13-34C045CEA7681
Url	String	发起核身流程的URL。 - 用于H5场景核身。 示例值：https://open.weixin.qq.com/connect/oauth2/authorize?appid=wxa50a8f2ac379585f53&redirect_uri=https%3A%2F%2Feid.faceid.qq.com%2Fapi%2Fv1%2FGetOpenId%3Ft 0F1E-455BD-BE13-34C05CEA7681&component_appid=wx9802ee81e68d6dee#wechat_redirec
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提 RequestId。

4. 示例

示例1 获取E证通Token成功示例

输入示例

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

{
"MerchantId": "0NSJ240626160443125212"
}
```

输出示例

```
{
"Response": {
"EidToken": "CE661F1A-0F1E-45BD-BE13-34C05CEA7681",
"RequestId": "f904f4cf-75db-4f8f-a5ec-dc4f942c7f7a",
"Url": "https://open.weixin.qq.com/connect/oauth2/authorize?appid=wxa50a8ac37958245f53&redirect_uri=https%3A%2F%2Feid.faceid.qq.com%2Fapi%2Fv1%2FGetOpenId%3Ftoken%3DCE3661F1A-0F1E-45BD-BE113-34C05CEA7681&response_type=code&scope=snsapi_base&state=&component_appid=wx9802ee81e68d6dee#wechat_redirect"
}
}
```

示例2 获取E证通Token失败示例

输入示例

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

{
"MerchantId": "0NSJ240615216044312521"
}
```

输出示例

```
{
"Response": {
"Error": {
"Code": "FailedOperation.UnregisteredEid",
"Message": "该用户未注册E证通，请先注册并跟公安库核验。"
},
"RequestId": "dc401e69-32bf-4002-83ca-3a12f62f2e4b"
}
}
```


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.DecryptSystemError	解密失败。
FailedOperation.StsUnAuthErrError	STS未授权。
FailedOperation.UnKnown	内部未知错误。
FailedOperation.UnregisteredEid	该用户未注册E证通，请先注册并跟权威库核验。
InvalidParameter	参数错误。
InvalidParameter.UnsupportEncryptField	存在不加密的字段，请参考文档修改。
InvalidParameterValue.RuleIdDisabled	该ruleid已被您停用，请确认后重试。
InvalidParameterValue.RuleIdNotExist	RuleId不存在，请到人脸核身控制台申请。
UnauthorizedOperation.Arrears	账号已欠费。
UnauthorizedOperation.NonAuthorize	账号未实名。
UnauthorizedOperation.Nonactivated	未开通服务。

获取SDK核验结果

最近更新时间：2025-05-26 19:47:47

1. 接口描述

接口请求域名： faceid.tencentcloudapi.com 。

完成验证后，用FaceIdToken调用本接口获取结果信息，FaceIdToken生成后三天内（3*24*3,600秒）可多次拉取。

默认接口请求频率限制：100次/秒。

推荐使用 API Explorer　点击调试

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

2. 输入参数

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

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：GetFaceIdResult。
Version	是	String	公共参数，本接口取值：2018-03-01。
Region	否	String	公共参数，本接口不需要传递此参数。
FaceIdToken	是	String	SDK人脸核身流程的标识。 - 调用GetFaceIdToken接口时生成。 示例值：CE6613F1A-0F1E-45BD-BE13-34C05CEA76181
IsNeedVideo	否	Boolean	是否需要拉取视频。 - 默认false：不需要。 示例值：false
IsNeedBestFrame	否	Boolean	是否需要拉取截帧。 - 默认false：不需要。 示例值：false
IsEncryptResponse	否	Boolean	是否对回包整体进行加密。 示例值：false
Encryption	否	Encryption	是否需要对返回中的敏感信息进行加密。 只需指定加密算法Algorithm即可，其余字段传入默认值。

3. 输出参数

参数名称	类型	描述
IdCard	String	身份证。 示例值：11204416541220243X
Name	String	姓名。 示例值：韦小宝
Result	String	业务核验结果。 - 参考：https://cloud.tencent.com/document/product/1007/47912。 示例值：0
Description	String	业务核验描述。 示例值：成功
Similarity	Float	相似度。 - 取值：0-100。

参数名称	类型	描述
		- 数值越大相似度越高。 示例值：97.7
VideoBase64	String	用户核验的视频base64。 - 如果选择了使用cos，返回完整cos地址，如https://bucket.cos.ap-guangzhou.myqcloud.com/objectKey。 注意：此字段可能返回 null，表示取不到有效值。 示例值：AAAAGGZ0eX...5Zin4btl4A
BestFrameBase64	String	用户核验视频的截帧base64。 - 如果选择了使用cos，返回完整cos地址如https://bucket.cos.ap-guangzhou.myqcloud.com/objectKey。 注意：此字段可能返回 null，表示取不到有效值。 示例值：/9j/4AAQSk...TYkEBjj//Z
Extra	String	获取token时透传的信息。 注意：此字段可能返回 null，表示取不到有效值。 示例值：20241101170759199020006000 0
DeviceInfoTag	String	plus版：描述当前请求所在设备的风险标签。 - 详情如下： 06-疑似黑产设备。 null-无设备风险。 - 增强版：此字段不生效，默认为null。 注意：此字段可能返回 null，表示取不到有效值。 示例值：06
RiskInfoTag	String	行为风险标签。 - 仅错误码返回1007（设备疑似被劫持）时返回风险标签。 - 标签说明： 02：攻击风险 注意：此字段可能返回 null，表示取不到有效值。 示例值：02
LivenessInfoTag	String	plus版：描述当前请求活体阶段被拒绝的详细原因。 - 详情如下： 01-用户全程闭眼。 02-用户未完成指定动作。 03-疑似翻拍攻击。 04-疑似合成图片。 05-疑似合成视频。 06-疑似合成动作。 07-疑似黑产模板。 08-疑似存在水印。 09-反光校验未通过。 10-最佳帧校验未通过。 11-人脸质量过差。 12-人脸距离不匹配。 13-疑似对抗样本攻击。 null-无。 - 增强版：此字段不生效，默认为null。 注意：此字段可能返回 null，表示取不到有效值。 示例值：01
DeviceInfoLevel	String	plus版：描述当前请求所在设备的风险等级，共4级。 - 详情如下： 1 - 安全。 2 - 低风险。 3 - 中风险。 4 - 高危。 null - 未获取到风险等级。 - 增强版：此字段不生效，默认为null。 注意：此字段可能返回 null，表示取不到有效值。 示例值：1
Encryption	Encryption	敏感数据加密信息。 注意：此字段可能返回 null，表示取不到有效值。
EncryptedBody	String	加密后的数据。 注意：此字段可能返回 null，表示取不到有效值。

参数名称	类型	描述
		示例值：0hMvx6fX6n...gXE0adJA==
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 获取SDK核验结果成功示例

输入示例

```
https://faceid.tencentcloudapi.com/?Action=GetFaceIdResult
&FaceIdToken=CE661F1A-0F1E-45BD-BE113-34C05CEA76811
&<公共请求参数>
```

输出示例

```
{
"Response": {
"BestFrameBase64": "/9j/4AAQSk...1Udr+R/9k=",
"Description": "成功",
"DeviceInfoLevel": null,
"DeviceInfoTag": null,
"Extra": "",
"IdCard": "",
"LivenessInfoTag": null,
"Name": "",
"RequestId": "13c51421-3b1d-4b40-8206-8dacedec2d32",
"Result": "0",
"RiskInfoTag": "",
"Similarity": 0,
"VideoBase64": ""
}
}
```

示例2 获取SDK核验结果人脸检测失败示例

输入示例

```
https://faceid.tencentcloudapi.com/?Action=GetFaceIdResult
&FaceIdToken=CE661F1A-0F1E-45BD-BE113-34C05CEA76831
&<公共请求参数>
```

输出示例

```
{
"Response": {
"BestFrameBase64": "/9j/4AAQSk...9Dl1t//9k=",
"Description": "人脸检测失败，无法提取比对照",
"DeviceInfoLevel": null,
"DeviceInfoTag": null,
"Extra": "",
"IdCard": "",
"LivenessInfoTag": null,
```

```
"Name": "",
"RequestId": "391e4084-43dd-4b03-b31c-c92a519d40bd",
"Result": "1004",
"RiskInfoTag": "",
"Similarity": 0,
"VideoBase64": "AAAAGGZ0eX...AAAAAAAAA="
}
}
```


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.UnKnown	内部未知错误。
InvalidParameter	参数错误。
InvalidParameterValue.BizTokenExpired	BizToken过期。
InvalidParameterValue.BizTokenIllegal	BizToken不合法。
UnauthorizedOperation.Arrears	账号已欠费。
UnauthorizedOperation.Nonactivated	未开通服务。

获取SDKToken

最近更新时间：2025-05-26 19:47:47

1. 接口描述

接口请求域名： faceid.tencentcloudapi.com 。

每次调用人脸核身SDK服务前，需先调用本接口获取SDKToken，用来串联核身流程，在验证完成后，用于获取验证结果信息，该token仅能核身一次。

默认接口请求频率限制：100次/秒。

推荐使用 API Explorer 点击调试

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

2. 输入参数

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

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：GetFaceIdToken。
Version	是	String	公共参数，本接口取值：2018-03-01。
Region	否	String	公共参数，本接口不需要传递此参数。
CompareLib	是	String	比对库。 - 取值范围： LOCAL：本地上传照片。 BUSINESS：商业库。 示例值：BUSINESS
IdCard	否	String	身份证。 - CompareLib为商业库时必传。 示例值：11204416541220243X
Name	否	String	姓名。 - CompareLib为商业库时必传。 示例值：韦小宝
ImageBase64	否	String	图片的Base64。 - CompareLib为上传照片比对时必传。 - Base64后图片最大8MB。 - 请使用标准的Base64编码方式(带=补位)，编码规范参考RFC4648。 示例值：/9j/4AAQSkZJRg.....s97n//2Q==
Meta	否	String	SDK中生成的Meta字符串。
Extra	否	String	透传参数。 - 1000长度字符串
UseCos	否	Boolean	是否使用cos桶。 - 默认为false。 - 设置该参数为true后，核身过程中的视频图片将会存储在人脸核身控制台授权cos的bucket中，拉取结果时会返回对应资源完整cos地址。 - 开通地址见https://console.cloud.tencent.com/faceid/cos -【注意】选择该参数为true后将不返回base64数据，请根据接入情况谨慎修改。 示例值：true
Encryption	否	Encryption	敏感数据加密信息。对传入信息（姓名、身份证号、自传照片）有加密需求的用户可使用此参数，详情请点击左侧链接。
RuleId	否	String	用于细分客户使用场景。 - 申请开通服务后，可以在腾讯云慧眼人脸核身控制台（https://console.cloud.tencent.com/faceid）自助接入里面创建，审核通过后即可调用。 - 如有疑问，请添加腾讯云人脸核身小助手进行咨询。 示例值：12

3. 输出参数

参数名称	类型	描述
FaceIdToken	String	token值。- 有效期 10分钟。- 只能完成1次核身。 示例值：42S4B131D-5974-4674-EABB-8FFF1EC99F189
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 获取 SDK 核验 Token成功示例

输入示例

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

{
"CompareLib": "BUSINESS",
"IdCard": "11204416541220243X",
"Name": "韦小宝"
}
```

输出示例

```
{
"Response": {
"FaceIdToken": "42S41B31D-5974-4674-EABB-8FFF1EC99F819",
"RequestId": "94b54cdf-d975-4718-b091-32f8d79d6397"
}
}
```

示例2 获取 SDK 核验 Token失败示例

输入示例

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

{
"CompareLib": "LOCAL",
"IdCard": "11204416541220243X",
"Name": "韦小宝",
"ImageBase64": "/9j/4AAQSkZJRg.....s97n//2Q=="
}
```

输出示例

```
{
"Response": {
"Error": {
"Code": "FailedOperation.ImageSizeTooLarge",
"Message": "图片尺寸过大。"
},
"RequestId": "ae8fbee0-f20e-4651-896e-e21e80d3822a"
}
}
```

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.DecryptSystemError	解密失败。
FailedOperation.ImageSizeTooLarge	图片尺寸过大。
FailedOperation.UnKnown	内部未知错误。
InvalidParameter	参数错误。
UnauthorizedOperation.Arrears	账号已欠费。
UnauthorizedOperation.Nonactivated	未开通服务。

实名信息核验相关接口
身份信息认证（二要素核验）

最近更新时间：2025-05-26 19:47:53

1. 接口描述

接口请求域名： faceid.tencentcloudapi.com 。

传入姓名和身份证号，校验两者的真实性和一致性。

默认接口请求频率限制：100次/秒。

推荐使用 API Explorer　点击调试

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

2. 输入参数

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

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：IdCardVerification。
Version	是	String	公共参数，本接口取值：2018-03-01。
Region	否	String	公共参数，本接口不需要传递此参数。
IdCard	是	String	身份证号。 示例值：112044165412202 43X
Name	是	String	姓名。 示例值：韦小宝
Encryption	否	Encryption	敏感数据加密信息。 - 对传入信息（姓名、身份证号）有加密需求的用户可使用此参数，详情请点击左侧链接。

3. 输出参数

参数名称	类型	描述
Result	String	认证结果码，收费情况如下。 - 收费结果码： 0: 姓名和身份证号一致 -1: 姓名和身份证号不一致 不收费结果码： -2: 非法身份证号（长度、校验位等不正确） -3: 非法姓名（长度、格式等不正确） -4: 证件库服务异常 -5: 证件库中无此身份证记录 -6: 权威比对系统升级中，请稍后再试 -7: 认证次数超过当日限制 示例值：0
Description	String	业务结果描述。 示例值：姓名和身份证号一致
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 身份信息认证一致示例

输入示例

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

{
"IdCard": "11204416541220243X",
"Name": "韦小宝"
}
```

输出示例

```
{
"Response": {
"Result": "0",
"Description": "姓名和身份证号一致",
"RequestId": "94b54cdf-d975-4718-b091-32f8d79d6397"
}
}
```

示例2 身份信息认证不一致示例

输入示例

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

{
"IdCard": "440305199505132561",
"Name": "刘洋"
}
```

输出示例

```
{
"Response": {
"Result": "-1",
"Description": "姓名和身份证号不一致",
"RequestId": "80c7abb8-4563-4636-98c3-0499f1611a33"
}
}
```

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.InvalidAuthorization	CAM签名/鉴权错误。
FailedOperation.DecryptSystemError	解密失败。
FailedOperation.StsUnAuthErrError	STS未授权。
FailedOperation.UnKnown	内部未知错误。
InvalidParameter	参数错误。
InvalidParameter.UnsupportEncryptField	存在不加密的字段，请参考文档修改。
InvalidParameterValue	参数取值错误。
UnauthorizedOperation.Arrears	账号已欠费。
UnauthorizedOperation.ChargeStatusException	计费状态异常。
UnauthorizedOperation.NonAuthorize	账号未实名。
UnauthorizedOperation.Nonactivated	未开通服务。

身份证识别及信息核验

最近更新时间：2025-05-26 19:47:53

1. 接口描述

接口请求域名： faceid.tencentcloudapi.com 。

本接口用于校验姓名和身份证号的真实性和一致性，您可以通过输入姓名和身份证号或传入身份证人像面照片提供所需验证信息。

默认接口请求频率限制：100次/秒。

推荐使用 API Explorer　点击调试

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

2. 输入参数

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

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：IdCardOCRVerification。
Version	是	String	公共参数，本接口取值：2018-03-01。
Region	否	String	公共参数，本接口不需要传递此参数。
IdCard	否	String	身份证号。 - 姓名和身份证号、ImageBase64、ImageUrl三者必须提供其中之一。 - 若都提供了，则按照姓名和身份证号>ImageBase64>ImageUrl的优先级使用参数。 示例值：11204416541220243X
Name	否	String	姓名。 示例值：韦小宝
ImageBase64	否	String	身份证人像面的 Base64 值。 - 支持的图片格式：PNG、JPG、JPEG，暂不支持 GIF 格式。 - 支持的图片大小：所下载图片经Base64编码后不超过 3M。请使用标准的Base64编码方式(带=补位)，编码规范参考RFC4648。 示例值：/9j/4AAQSkZJRg.....s97n//2Q==
ImageUrl	否	String	身份证人像面的 Url 地址。 - 支持的图片格式：PNG、JPG、JPEG，暂不支持 GIF 格式。 - 支持的图片大小：所下载图片经 Base64 编码后不超过 3M。图片下载时间不超过 3 秒。 - 图片存储于腾讯云的 Url 可保障更高的下载速度和稳定性，建议图片存储于腾讯云。 - 非腾讯云存储的 Url 速度和稳定性可能受一定影响。 示例值：https://www.qq.com/image.jpg
Encryption	否	Encryption	敏感数据加密信息。 - 对传入信息（姓名、身份证号）有加密需求的用户可使用此参数，详情请点击左侧链接。

3. 输出参数

参数名称	类型	描述
Result	String	认证结果码，收费情况如下。 - 收费结果码： 0: 姓名和身份证号一致。 -1: 姓名和身份证号不一致。 - 不收费结果码： -2: 非法身份证号（长度、校验位等不正确）。 -3: 非法姓名（长度、格式等不正确）。 -4: 证件库服务异常。 -5: 证件库中无此身份证记录。 -6: 权威比对系统升级中，请稍后再试。

参数名称	类型	描述
		-7: 认证次数超过当日限制。 示例值：0
Description	String	业务结果描述。 示例值：姓名和身份证号一致
Name	String	用于验证的姓名。 示例值：韦小宝
IdCard	String	用于验证的身份证号。 示例值：112044165412202 43X
Sex	String	OCR得到的性别。 注意：此字段可能返回 null，表示取不到有效值。 示例值：男
Nation	String	OCR得到的民族。 注意：此字段可能返回 null，表示取不到有效值。 示例值：汉
Birth	String	OCR得到的生日。 注意：此字段可能返回 null，表示取不到有效值。 示例值：1654/12/20
Address	String	OCR得到的地址。 注意：此字段可能返回 null，表示取不到有效值。 示例值：北京市东城区景山前街4号紫禁城敬事房
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 使用照片URL进行核验示例

输入示例

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

{
"ImageUrl": "https://www.qq.com/image.jpg"
}
```

输出示例

```
{
"Response": {
"Result": "0",
"Description": "姓名和身份证号一致",
"Name": "韦小宝",
"IdCard": "112044165412202 43X",
"Sex": "男",
"Nation": "汉",
"Birth": "1654/12/20",
"Address": "北京市东城区景山前街4号紫禁城敬事房",
"RequestId": "a62f567c-1eea-4ef3-b51a-a9eb9bd84cd9"
```

```
}
}
```

示例2 使用照片Base64进行核验示例

输入示例

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

{
"ImageBase64": "/9j/4AAQSkZJRg.....s97n//2Q=="
}
```

输出示例

```
{
"Response": {
"Result": "0",
"Description": "姓名和身份证号一致",
"Name": "韦小宝",
"IdCard": "11204416541220243X",
"Sex": "男",
"Nation": "汉",
"Birth": "1654/12/20",
"Address": "北京市东城区景山前街4号紫禁城敬事房",
"RequestId": "022ffdd2-67a2-4177-8946-97bc1c4b3347"
}
}
```

示例3 使用姓名身份证号进行验证示例

输入示例

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

{
"IdCard": "11204416541220243X",
"Name": "韦小宝"
}
```

输出示例

```
{
"Response": {
"Result": "0",
"Description": "姓名和身份证号一致",
"Name": "韦小宝",
```

```
"IdCard": "11204416541220243X",
"Sex": "",
"Nation": "",
"Birth": "",
"Address": "",
"RequestId": "945c69ad-d86c-47ea-ba33-419b1dc4d242"
}
}
```

示例4 身份证识别及信息核验失败示例

输入示例

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

{
"IdCard": "11204416541220243Y",
"Name": "韦小宝"
}
```

输出示例

```
{
"Response": {
"Address": "",
"Birth": "",
"Description": "非法身份证号（长度、校验位等不正确）",
"IdCard": "11204416541220243Y",
"Name": "韦小宝",
"Nation": "",
"RequestId": "c528602c-6574-4dd7-a836-895ff62a13ed",
"Result": "-2",
"Sex": ""
}
}
```

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.InvalidAuthorization	CAM签名/鉴权错误。
FailedOperation.DecryptSystemError	解密失败。
FailedOperation.DownLoadError	文件下载失败。
FailedOperation.EmptyImageError	图片内容为空。
FailedOperation.ImageBlur	图片模糊。
FailedOperation.ImageDecodeFailed	图片解码失败。
FailedOperation.ImageNoIdCard	图片中未检测到身份证。
FailedOperation.ImageSizeTooLarge	图片尺寸过大。
FailedOperation.OcrFailed	Ocr识别失败。
FailedOperation.StsUnAuthErrError	STS未授权。
FailedOperation.UnKnown	内部未知错误。
InvalidParameter	参数错误。
InvalidParameter.UnsupportEncryptField	存在不加密的字段，请参考文档修改。
InvalidParameterValue	参数取值错误。
UnauthorizedOperation.Arrears	账号已欠费。
UnauthorizedOperation.ChargeStatusException	计费状态异常。
UnauthorizedOperation.NonAuthorize	账号未实名。
UnauthorizedOperation.Nonactivated	未开通服务。

身份信息及有效期核验

最近更新时间：2025-05-26 19:47:54

1. 接口描述

接口请求域名： faceid.tencentcloudapi.com 。

本接口用于校验姓名、身份证号、身份证有效期的真实性和一致性。

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer 点击调试

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

2. 输入参数

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

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：CheckIdNameDate。
Version	是	String	公共参数，本接口取值：2018-03-01。
Region	否	String	公共参数，本接口不需要传递此参数。
Name	是	String	姓名。 示例值：韦小宝
IdCard	是	String	身份证号。 示例值：112044165412202 43X
ValidityBegin	是	String	身份证有效期开始时间。 - 格式：YYYYMMDD，如：20210701。 示例值：20160607
ValidityEnd	是	String	身份证有效期到期时间。 格式：YYYYMMDD，长期用“00000000”代替，如：20210701。 示例值：20250607
Encryption	否	Encryption	敏感数据加密信息。 - 对传入信息（姓名、身份证号）有加密需求的用户可使用此参数，详情请点击左侧链接。

3. 输出参数

参数名称	类型	描述
Result	String	认证结果码，收费情况如下。 - 收费结果码： 0: 一致。 -1: 不一致。 - 不收费结果码： -2: 非法身份证号（长度、校验位等不正确）。 -3: 非法姓名（长度、格式等不正确）。 -4: 非法有效期（长度、格式等不正确）。 -5: 身份信息无效。 -6: 证件库服务异常。 -7: 证件库中无此身份证记录。 -8: 认证次数超过当日限制，请次日重试。 示例值："0"

参数名称	类型	描述
Description	String	业务结果描述。 示例值：一致
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 身份信息及有效期核验成功示例

输入示例

```
https://faceid.tencentcloudapi.com/?Action=CheckIdNameDate
&Name=韦小宝
&IdCard= 11204416541220243X
&ValidityBegin=20160204
&ValidityEnd=20260204
&<公共请求参数>
```

输出示例

```
{
"Response": {
"Result": "0",
"Description": "一致",
"RequestId": "8695c53f-776f-4ff5-a66d-84e67b352d20"
}
}
```

示例2 身份信息及有效期核验不一致示例

输入示例

```
https://faceid.tencentcloudapi.com/?Action=CheckIdNameDate
&Name=韦小宝
&IdCard= 11204416541220243X
&ValidityBegin=20160204
&ValidityEnd=20260204
&<公共请求参数>
```

输出示例

```
{
"Response": {
"Description": "不一致",
"RequestId": "af4075de-427f-48a5-85cb-2be5b3751b76",
"Result": "-1"
}
}
```

5. 开发者资源

腾讯云 API 平台

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

API Inspector

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

SDK

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

- Tencent Cloud SDK 3.0 for Python: GitHub, Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub, Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub, Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub, Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub, Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub, Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub, Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub, Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

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.DecryptSystemError	解密失败。
FailedOperation.StsUnAuthErrError	STS未授权。
InternalError	内部错误。
InvalidParameter.UnsupportEncryptField	存在不加密的字段，请参考文档修改。
InvalidParameterValue	参数取值错误。
UnauthorizedOperation.Arrears	账号已欠费。
UnauthorizedOperation.ChargeStatusException	计费状态异常。
UnauthorizedOperation.NonAuthorize	账号未实名。
UnauthorizedOperation.Nonactivated	未开通服务。

银行卡基础信息查询

最近更新时间：2025-06-12 01:35:50

1. 接口描述

接口请求域名： faceid.tencentcloudapi.com 。

银行卡基础信息查询

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：CheckBankCardInformation。
Version	是	String	公共参数，本接口取值：2018-03-01。
Region	否	String	公共参数，本接口不需要传递此参数。
BankCard	是	String	银行卡号。 示例值：6225768888888888
Encryption	否	Encryption	敏感数据加密信息。 - 对传入信息（银行卡号）有加密需求的用户可使用此参数，详情请点击左侧链接。

3. 输出参数

参数名称	类型	描述
Result	String	认证结果码，收费情况如下。 - 收费结果码： 0: 查询成功 -1: 未查到信息 -5: 卡号无效 - 不收费结果码： -2：验证中心服务繁忙 -3：银行卡不存在 -4：认证次数超过当日限制，请次日重试 -6: 暂不支持该银行卡种 示例值：0
Description	String	业务结果描述。 示例值：查询成功
AccountBank	String	开户行。 示例值：中国工商银行
AccountType	Integer	卡性质。 - 取值范围： 1：借记卡。 2：贷记卡。 3：预付费卡。 4：准贷记卡 示例值：1

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

4. 示例

示例1 银行卡基础信息查询成功示例

输入示例

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

{
"BankCard": "6225768888888888"
}
```

输出示例

```
{
"Response": {
"Result": "0",
"Description": "查询成功",
"AccountBank": "招商银行信用卡",
"AccountType": 2,
"RequestId": "8695c53f-776f-4ff5-a66d-84e67b352d20"
}
}
```

示例2 银行卡基础信息查询失败示例

输入示例

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

{
"BankCard": "6225768888888888"
}
```

输出示例

```
{
"Response": {
"AccountBank": "",
"AccountType": -1,
"Description": "未查到信息",
"RequestId": "bd41a241-0e50-4337-999e-0687d56a34c8",
"Result": "-1"
```

```
}
}
```

5. 开发者资源

腾讯云 API 平台

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

API Inspector

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

SDK

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

- Tencent Cloud SDK 3.0 for Python: GitHub, Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub, Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub, Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub, Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub, Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub, Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub, Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub, Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

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.InvalidAuthorization	CAM签名/鉴权错误。
FailedOperation.DecryptSystemError	解密失败。
FailedOperation.StsUnAuthErrError	STS未授权。
FailedOperation.UnKnown	内部未知错误。
InternalError	内部错误。
InvalidParameter.UnsupportEncryptField	存在不加密的字段，请参考文档修改。
UnauthorizedOperation.Arrears	账号已欠费。
UnauthorizedOperation.ChargeStatusException	计费状态异常。
UnauthorizedOperation.NonAuthorize	账号未实名。
UnauthorizedOperation.Nonactivated	未开通服务。

银行卡二要素核验

最近更新时间：2025-06-12 01:35:50

1. 接口描述

接口请求域名： faceid.tencentcloudapi.com 。

本接口用于校验姓名和银行卡号的真实性和一致性。

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer 点击调试

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

2. 输入参数

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

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：BankCard2EVerification。
Version	是	String	公共参数，本接口取值：2018-03-01。
Region	否	String	公共参数，本接口不需要传递此参数。
Name	是	String	姓名。 示例值：韦小宝
BankCard	是	String	银行卡。 示例值：6225768888888888
Encryption	否	Encryption	敏感数据加密信息。 - 对传入信息（姓名、银行卡号）有加密需求的用户可使用此参数，详情请点击左侧链接。

3. 输出参数

参数名称	类型	描述
Result	String	认证结果码。 - 计费结果码： '0': '认证通过'。 '-1': '认证未通过'。 '-4': '持卡人信息有误'。 '-5': '未开通无卡支付'。 '-6': '此卡被没收'。 '-7': '无效卡号'。 '-8': '此卡无对应发卡行'。 '-9': '该卡未初始化或睡眠卡'。 '-10': '作弊卡、吞卡'。 '-11': '此卡已挂失'。 '-12': '该卡已过期'。 '-13': '受限制的卡'。 '-14': '密码错误次数超限'。 '-15': '发卡行不支持此交易'。 '-18': '卡状态异常或卡号错误'。 - 不计费结果码： '-2': '姓名校验不通过'。 '-3': '银行卡号格式有误'。 '-16': '验证中心服务繁忙'。 '-17': '验证次数超限，请次日重试'。

参数名称	类型	描述
		示例值：0
Description	String	业务结果描述。 示例值：认证通过
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 银行卡二要素核验一致示例

输入示例

```
https://faceid.tencentcloudapi.com/?Action=BankCard2EVerification
&Name=韦小宝
&BankCard= 6225768888888888
&<公共请求参数>
```

输出示例

```
{
"Response": {
"Result": "0",
"Description": "认证通过",
"RequestId": "c6daaf7f-dbdc-4a9d-a20b-9a14ffdd8328"
}
}
```

示例2 银行卡二要素核验不一致示例

输入示例

```
https://faceid.tencentcloudapi.com/?Action=BankCard2EVerification
&Name=韦小宝
&BankCard= 6226090210146748
&<公共请求参数>
```

输出示例

```
{
"Response": {
"Result": "-4",
"Description": "持卡人信息有误",
"RequestId": "d668328c-7847-42d7-bdce-215ebadffd9b"
}
}
```

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.DecryptSystemError	解密失败。
FailedOperation.StsUnAuthErrError	STS未授权。
FailedOperation.UnKnown	内部未知错误。
InvalidParameter	参数错误。
InvalidParameter.UnsupportEncryptField	存在不加密的字段，请参考文档修改。
InvalidParameterValue	参数取值错误。
UnauthorizedOperation.Arrears	账号已欠费。
UnauthorizedOperation.ChargeStatusException	计费状态异常。
UnauthorizedOperation.NonAuthorize	账号未实名。
UnauthorizedOperation.Nonactivated	未开通服务。

银行卡三要素核验

最近更新时间：2025-06-12 01:35:50

1. 接口描述

接口请求域名： faceid.tencentcloudapi.com 。

本接口用于银行卡号、姓名、开户证件号信息的真实性和一致性。

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer 点击调试

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

2. 输入参数

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

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：BankCardVerification。
Version	是	String	公共参数，本接口取值：2018-03-01。
Region	否	String	公共参数，本接口不需要传递此参数。
IdCard	是	String	开户证件号。 - 与CertType参数的证件类型一致，如：身份证，则传入身份证号。 示例值：440111201903211111
Name	是	String	姓名。 示例值：韦小宝
BankCard	是	String	银行卡。 示例值：6225768888888888
CertType	否	Integer	证件类型。 - 请确认该证件为开户时使用的证件类型，未用于开户的证件信息不支持验证。 - 目前默认：0 身份证，其他证件类型暂不支持。 示例值：0
Encryption	否	Encryption	敏感数据加密信息。 - 对传入信息（姓名、身份证号、银行卡号）有加密需求的用户可使用此参数，详情请点击左侧链接。

3. 输出参数

参数名称	类型	描述
Result	String	认证结果码。 - 收费结果码： '0': '认证通过'。 '-1': '认证未通过'。 '-5': '持卡人信息有误'。 '-6': '未开通无卡支付'。 '-7': '此卡被没收'。 '-8': '无效卡号'。 '-9': '此卡无对应发卡行'。 '-10': '该卡未初始化或睡眠卡'。 '-11': '作弊卡、吞卡'。 '-12': '此卡已挂失'。 '-13': '该卡已过期'。 '-14': '受限制的卡'。 '-15': '密码错误次数超限'。 '-16': '发卡行不支持此交易'。

参数名称	类型	描述
		'-20': '卡状态异常或卡号错误'。 - 不收费结果码： '-2': '姓名校验不通过'。 '-3': '身份证号码有误'。 '-4': '银行卡号格式有误'。 '-17': '验证中心服务繁忙'。 '-18': '验证次数超限，请次日重试'。 '-19': '该证件号暂不支持核验，当前仅支持二代身份证'。 示例值：0
Description	String	业务结果描述。 示例值：认证通过
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 银行卡三要素核验一致示例

输入示例

```
https://faceid.tencentcloudapi.com/?Action=BankCardVerification
&IdCard=11204416541220243X
&Name=韦小宝
&BankCard= 6226090210146748
&<公共请求参数>
```

输出示例

```
{
"Response": {
"Result": "0",
"Description": "认证通过",
"RequestId": "a5fdb909-5ee6-43ff-a152-bb1b9744ee8b"
}
}
```

示例2 银行卡三要素核验不一致示例

输入示例

```
https://faceid.tencentcloudapi.com/?Action=BankCardVerification
&IdCard= 11204416541220243X
&Name=韦小宝
&BankCard= 6225768888888888
&<公共请求参数>
```

输出示例

```
{
"Response": {
"Result": "-1",
"Description": "认证未通过",
"RequestId": "89f695b2-18fd-44b6-bffc-96972052666f"
```

```
}
}
```

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.DecryptSystemError	解密失败。
FailedOperation.StsUnAuthErrError	STS未授权。
FailedOperation.UnKnown	内部未知错误。
InvalidParameter	参数错误。
InvalidParameter.UnsupportEncryptField	存在不加密的字段，请参考文档修改。
InvalidParameterValue	参数取值错误。
UnauthorizedOperation.Arrears	账号已欠费。
UnauthorizedOperation.ChargeStatusException	计费状态异常。
UnauthorizedOperation.NonAuthorize	账号未实名。
UnauthorizedOperation.Nonactivated	未开通服务。

银行卡四要素核验

最近更新时间：2025-06-12 01:35:50

1. 接口描述

接口请求域名： faceid.tencentcloudapi.com 。

本接口用于输入银行卡号、姓名、开户证件号、开户手机号，校验信息的真实性和一致性。

默认接口请求频率限制：20次/秒。

推荐使用 API Explorer　点击调试

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

2. 输入参数

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

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：BankCard4EVerification。
Version	是	String	公共参数，本接口取值：2018-03-01。
Region	否	String	公共参数，本接口不需要传递此参数。
Name	是	String	姓名。 示例值：韦小宝
BankCard	是	String	银行卡。 示例值：6225768888888888
Phone	是	String	手机号码。 示例值：16137688175
IdCard	是	String	开户证件号。 - 与CertType参数的证件类型一致，如：身份证，则传入身份证号。 示例值：112044165412202 43X
CertType	否	Integer	证件类型。 - 请确认该证件为开户时使用的证件类型，未用于开户的证件信息不支持验证。 - 目前默认为0：身份证，其他证件类型暂不支持。 示例值：0
Encryption	否	Encryption	敏感数据加密信息。 - 对传入信息（姓名、身份证号、手机号、银行卡号）有加密需求的用户可使用此参数，详情请点击左侧链接。

3. 输出参数

参数名称	类型	描述
Result	String	认证结果码。 - 收费结果码： '0': '认证通过'。 '-1': '认证未通过'。 '-6': '持卡人信息有误'。 '-7': '未开通无卡支付'。 '-8': '此卡被没收'。 '-9': '无效卡号'。 '-10': '此卡无对应发卡行'。 '-11': '该卡未初始化或睡眠卡'。 '-12': '作弊卡、吞卡'。 '-13': '此卡已挂失'。 '-14': '该卡已过期'。

参数名称	类型	描述
		'-15': '受限制的卡'。 '-16': '密码错误次数超限'。 '-17': '发卡行不支持此交易'。 '-21': '卡状态异常或卡号错误'。 - 不收费结果码： '-2': '姓名校验不通过'。 '-3': '身份证号码有误'。 '-4': '银行卡号格式有误'。 '-5': '手机号码不合法'。 '-18': '验证中心服务繁忙'。 '-19': '验证次数超限，请次日重试'。 '-20': '该证件号暂不支持核验，当前仅支持二代身份证'。 示例值：0
Description	String	业务结果描述。 示例值：认证通过
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 银行卡四要素核验一致示例

输入示例

```
https://faceid.tencentcloudapi.com/?Action=BankCard4EVerification
&Name=韦小宝
&BankCard=6225768888888888
&Phone=16137688175
&IdCard=11204416541220243X
&<公共请求参数>
```

输出示例

```
{
"Response": {
"Result": "0",
"Description": "认证通过",
"RequestId": "74e742e8-f91d-49c3-9744-c20f3baca117"
}
}
```

示例2 银行卡四要素核验不一致示例

输入示例

```
https://faceid.tencentcloudapi.com/?Action=BankCard4EVerification
&Name=韦小宝
&BankCard=6226090210146748
&Phone=16137688175
&IdCard= 11204416541220243X
&<公共请求参数>
```

输出示例

```
{
"Response": {
"Result": "-6",
"Description": "持卡人信息有误",
"RequestId": "24fe7851-49e9-4a4a-ac1e-3bd5c09323fd"
}
}
```

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.DecryptSystemError	解密失败。
FailedOperation.StsUnAuthErrError	STS未授权。
FailedOperation.UnKnown	内部未知错误。
InvalidParameter	参数错误。
InvalidParameter.UnsupportEncryptField	存在不加密的字段，请参考文档修改。
InvalidParameterValue	参数取值错误。
UnauthorizedOperation.Arrears	账号已欠费。
UnauthorizedOperation.ChargeStatusException	计费状态异常。
UnauthorizedOperation.NonAuthorize	账号未实名。
UnauthorizedOperation.Nonactivated	未开通服务。

手机号在网时长核验

最近更新时间：2025-05-26 19:47:53

1. 接口描述

接口请求域名： faceid.tencentcloudapi.com 。

本接口用于查询手机号在网时长，输入手机号进行查询。

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：MobileNetworkTimeVerification。
Version	是	String	公共参数，本接口取值：2018-03-01。
Region	否	String	公共参数，本接口不需要传递此参数。
Mobile	是	String	手机号码。 示例值：16137688175
Encryption	否	Encryption	敏感数据加密信息。 - 对传入信息（手机号）有加密需求的用户可使用此参数，详情请点击左侧链接。

3. 输出参数

参数名称	类型	描述
Result	String	认证结果码，收费情况如下。 - 收费结果码： 0: 成功。 -2: 手机号不存在。 -3: 手机号存在，但无法查询到在网时长。 - 不收费结果码： -1: 手机号格式不正确。 -4: 验证中心服务繁忙。 -5：认证次数超过当日限制，请次日重试。 示例值：0
Description	String	业务结果描述。 示例值：成功
Range	String	在网时长区间。 - 格式为[a,b)，表示在网时长在a个月以上，b个月以下。 - 若b为+时表示没有上限。 示例值：[23,+)
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 手机号在网时长核验成功示例

输入示例

```
https://faceid.tencentcloudapi.com/?Action=MobileNetworkTimeVerification
&Mobile=13800138000
&<公共请求参数>
```

输出示例

```
{
"Response": {
"Result": "0",
"Description": "成功",
"Range": "[24,+)",
"RequestId": "f893bfcf-167d-45df-99aa-60a23fe5809d"
}
}
```

示例2 手机号在网时长核验异常示例

输入示例

```
https://faceid.tencentcloudapi.com/?Action=MobileNetworkTimeVerification
&Mobile=16137688175
&<公共请求参数>
```

输出示例

```
{
"Response": {
"Result": "-2",
"Description": "手机号不存在",
"Range": "",
"RequestId": "3151331a-277e-4317-891d-0ef4e0afdd3e"
}
}
```

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.DecryptSystemError	解密失败。
FailedOperation.StsUnAuthErrError	STS未授权。
FailedOperation.UnKnown	内部未知错误。
InvalidParameter	参数错误。
InvalidParameter.UnsupportEncryptField	存在不加密的字段，请参考文档修改。
InvalidParameterValue	参数取值错误。
InvalidParameterValue.BizTokenExpired	BizToken过期。
InvalidParameterValue.RuleIdNotExist	RuleId不存在，请到人脸核身控制台申请。
UnauthorizedOperation.Arrears	账号已欠费。
UnauthorizedOperation.ChargeStatusException	计费状态异常。
UnauthorizedOperation.NonAuthorize	账号未实名。
UnauthorizedOperation.Nonactivated	未开通服务。

手机号状态查询

最近更新时间：2025-05-26 19:47:53

1. 接口描述

接口请求域名： faceid.tencentcloudapi.com 。

本接口用于验证手机号的状态，您可以输入手机号进行查询。

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：MobileStatus。
Version	是	String	公共参数，本接口取值：2018-03-01。
Region	否	String	公共参数，本接口不需要传递此参数。
Mobile	是	String	手机号码。 示例值：16137688175
Encryption	否	Encryption	敏感数据加密信息。 - 对传入信息（手机号）有加密需求的用户可使用此参数，详情请点击左侧链接。

3. 输出参数

参数名称	类型	描述
Result	String	认证结果码，收费情况如下。 - 收费结果码： 0：成功。 - 不收费结果码： -1：未查询到结果。 -2：手机号格式不正确。 -3：验证中心服务繁忙。 -4：认证次数超过当日限制，请次日重试。 示例值：0
Description	String	业务结果描述。 示例值：成功
StatusCode	Integer	状态码。 - 取值范围： 0：正常。 1：停机。 2：销号。 4：不在网。 99：未知状态。 示例值：1
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 手机号状态查询成功示例

输入示例

```
https://faceid.tencentcloudapi.com/?Action=MobileStatus
&Mobile=13800138000
&<公共请求参数>
```

输出示例

```
{
"Response": {
"Result": "0",
"Description": "成功",
"StatusCode": 1,
"RequestId": "f893bfcf-167d-45df-99aa-60a23fe5809d"
}
}
```

示例2 手机号状态查询异常示例

输入示例

```
https://faceid.tencentcloudapi.com/?Action=MobileStatus
&Mobile=16137688175
&<公共请求参数>
```

输出示例

```
{
"Response": {
"Description": "手机号格式不正确",
"RequestId": "7411a5fc-0ba6-431d-9e8a-5f6537fb8701",
"Result": "-2",
"StatusCode": 99
}
}
```

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.DecryptSystemError	解密失败。
FailedOperation.StsUnAuthErrError	STS未授权。
FailedOperation.UnKnown	内部未知错误。
InvalidParameter	参数错误。
InvalidParameter.UnsupportEncryptField	存在不加密的字段，请参考文档修改。
InvalidParameterValue	参数取值错误。
UnauthorizedOperation.Arrears	账号已欠费。
UnauthorizedOperation.ChargeStatusException	计费状态异常。
UnauthorizedOperation.NonAuthorize	账号未实名。
UnauthorizedOperation.Nonactivated	未开通服务。

手机号三要素核验

最近更新时间：2025-05-26 19:47:46

1. 接口描述

接口请求域名： faceid.tencentcloudapi.com 。

本接口用于校验手机号、姓名和身份证号的真实性和一致性。支持的手机号段详情请查阅运营商类文档。

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：PhoneVerification。
Version	是	String	公共参数，本接口取值：2018-03-01。
Region	否	String	公共参数，本接口不需要传递此参数。
IdCard	是	String	身份证号。 示例值：11204416541220243X
Name	是	String	姓名。 示例值：韦小宝
Phone	是	String	手机号。 示例值：16137688175
VerifyMode	否	String	验证模式（详版/简版）。 - 简版与详版价格不一致，详见价格说明。 - 枚举值：0（简版）；1（详版）。 - 默认值为0。 示例值：0
CiphertextBlob	否	String	有加密需求的用户，传入kms的CiphertextBlob。关于数据加密可查阅 数据加密 文档。 示例值：BKD5+E/euB3mXQ9pqKNLvazmrSB/XdTsOasrx+m.....UY7JJ7p/WTFirTINnU4=
EncryptList.N	否	Array of String	在使用加密服务时，填入要被加密的字段。 - 本接口中可填入加密后的IdCard，Name，Phone中的一个或多个。 示例值：[IdCard]
Iv	否	String	有加密需求的用户，传入CBC加密的初始向量。 示例值：+am/h1FrULT8Y39M

3. 输出参数

参数名称	类型	描述
Result	String	认证结果码。 - 收费结果码 0: 三要素信息一致。 -4: 三要素信息不一致。 - 不收费结果码 -6: 手机号码不合法。 -7: 身份证号码有误。 -8: 姓名校验不通过。 -9: 没有记录。

参数名称	类型	描述
		-11: 验证中心服务繁忙。 -12: 认证次数超过当日限制，请次日重试。 示例值：0
Description	String	业务结果描述。 示例值：认证通过
Isp	String	运营商名称。 - 取值范围为["","移动","电信","联通"] 示例值：移动
ResultDetail	String	业务结果详细信息。 - 当VerifyMode配置"详版"，且Result为"-4: 三要素信息不一致"时返回。 - 枚举值： PhoneIdCardMismatch：手机号码与姓名一致，与身份证号不一致。 PhoneNameMismatch：手机号码身份证号一致，与姓名不一致。 PhoneNameIdCardMismatch：手机号码与姓名和身份证号均不一致。 NameIdCardMismatch：姓名和身份证号不一致。 OtherMismatch：其他不一致。 示例值：PhoneIdCardMismatch
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 手机号三要素核验一致示例

输入示例

```
https://faceid.tencentcloudapi.com/?Action=PhoneVerification
&IdCard=11204416541220243X
&Name=韦小宝
&Phone=16137688175
&<公共请求参数>
```

输出示例

```
{
"Response": {
"Result": "0",
"Description": "认证通过",
"Isp": "电信",
"RequestId": "a5fdb909-5ee6-43ff-a152-bb1b9744ee8b"
}
}
```

示例2 手机号三要素核验不一致示例

输入示例

```
https://faceid.tencentcloudapi.com/?Action=PhoneVerification
&IdCard=11204416541220243X
&Name=韦小宝
&Phone=16137688175
&<公共请求参数>
```

输出示例

```
{
"Response": {
"Description": "信息不一致",
"Isp": "联通",
"RequestId": "884a35af-289f-4b4e-a0b3-2315f02ab31e",
"Result": "-4"
}
}
```


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.DecryptSystemError	解密失败。
FailedOperation.StsUnAuthErrError	STS未授权。
FailedOperation.UnKnown	内部未知错误。
InvalidParameter	参数错误。
InvalidParameter.UnsupportEncryptField	存在不加密的字段，请参考文档修改。
InvalidParameterValue	参数取值错误。
UnauthorizedOperation.Arrears	账号已欠费。
UnauthorizedOperation.ChargeStatusException	计费状态异常。
UnauthorizedOperation.NonAuthorize	账号未实名。
UnauthorizedOperation.Nonactivated	未开通服务。

手机号三要素核验（移动）

最近更新时间：2025-05-26 19:47:52

1. 接口描述

接口请求域名： faceid.tencentcloudapi.com 。

本接口用于校验中国移动手机号、姓名和身份证号的真实性和一致性。中国移动支持的手机号段详情请查阅运营商类文档。

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：PhoneVerificationCMCC。
Version	是	String	公共参数，本接口取值：2018-03-01。
Region	否	String	公共参数，本接口不需要传递此参数。
IdCard	是	String	身份证号。 示例值：11204416541220243X
Name	是	String	姓名。 示例值：韦小宝
Phone	是	String	手机号。 示例值：16137688175
Encryption	否	Encryption	敏感数据加密信息。 - 对传入信息（姓名、身份证号、手机号）有加密需求的用户可使用此参数，详情请点击左侧链接。

3. 输出参数

参数名称	类型	描述
Result	String	认证结果码，收费情况如下。 - 收费结果码： 0: 认证通过。 -4: 信息不一致（手机号已实名，但姓名和身份证号与实名信息不一致）。 - 不收费结果码： -6: 手机号码不合法。 -7: 身份证号码有误。 -8: 姓名校验不通过。 -9: 没有记录。 -11: 验证中心服务繁忙。 示例值：0
Isp	String	运营商名称。 - 取值范围为["移动","联通","电信",""]。 示例值：移动
Description	String	业务结果描述。 示例值：业务结果描述。
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 移动手机号三要素核验一致示例

输入示例

```
https://faceid.tencentcloudapi.com/?Action=PhoneVerificationCMCC
&IdCard=11204416541220243X
&Name=韦小宝
&Phone=16137688175
&<公共请求参数>
```

输出示例

```
{
"Response": {
"Result": "0",
"Description": "认证通过",
"Isp": "移动",
"RequestId": "a5fdb909-5ee6-43ff-a152-bb1b9744ee8b"
}
}
```

示例2 移动手机号三要素核验不一致示例

输入示例

```
https://faceid.tencentcloudapi.com/?Action=PhoneVerificationCMCC
&IdCard=11204416541220243X
&Name=韦小宝
&Phone=16137688175
&<公共请求参数>
```

输出示例

```
{
"Response": {
"Description": "信息不一致",
"Isp": "移动",
"RequestId": "e281fdaa-6a24-4d37-82a3-acf28bc6c51f",
"Result": "-4"
}
}
```

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.DecryptSystemError	解密失败。
FailedOperation.StsUnAuthErrError	STS未授权。
FailedOperation.UnKnown	内部未知错误。
InvalidParameter	参数错误。
InvalidParameter.UnsupportEncryptField	存在不加密的字段，请参考文档修改。
InvalidParameterValue	参数取值错误。
UnauthorizedOperation.Arrears	账号已欠费。
UnauthorizedOperation.ChargeStatusException	计费状态异常。
UnauthorizedOperation.NonAuthorize	账号未实名。
UnauthorizedOperation.Nonactivated	未开通服务。

手机号三要素核验（电信）

最近更新时间：2025-05-26 19:47:52

1. 接口描述

接口请求域名： faceid.tencentcloudapi.com 。

本接口用于校验中国电信手机号、姓名和身份证号的真实性和一致性。中国电信支持的手机号段详情请查阅运营商类文档。

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：PhoneVerificationCTCC。
Version	是	String	公共参数，本接口取值：2018-03-01。
Region	否	String	公共参数，本接口不需要传递此参数。
IdCard	是	String	身份证号。 示例值：11204416541220243X
Name	是	String	姓名。 示例值：韦小宝
Phone	是	String	手机号。 示例值：16137688175
Encryption	否	Encryption	敏感数据加密信息。 - 对传入信息（姓名、身份证号、手机号）有加密需求的用户可使用此参数，详情请点击左侧链接。

3. 输出参数

参数名称	类型	描述
Result	String	认证结果码，收费情况如下。 - 收费结果码： 0: 认证通过。 -4: 信息不一致（手机号已实名，但姓名和身份证号与实名信息不一致）。 - 不收费结果码： -6: 手机号码不合法。 -7: 身份证号码有误。 -8: 姓名校验不通过。 -9: 没有记录。 -11: 验证中心服务繁忙。 示例值：0
Isp	String	运营商名称。 - 取值范围为["移动","联通","电信",""]。 示例值：电信
Description	String	业务结果描述。 示例值：比对一致
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 电信手机号三要素核验一致示例

输入示例

```
https://faceid.tencentcloudapi.com/?Action=PhoneVerificationCTCC
&IdCard=11204416541220243X
&Name=韦小宝
&Phone=16137688175
&<公共请求参数>
```

输出示例

```
{
"Response": {
"Result": "0",
"Description": "认证通过",
"Isp": "电信",
"RequestId": "a5fdb909-5ee6-43ff-a152-bb1b9744ee8b"
}
}
```

示例2 电信手机号三要素核验不一致示例

输入示例

```
https://faceid.tencentcloudapi.com/?Action=PhoneVerificationCTCC
&IdCard=11204416541220243X
&Name=韦小宝
&Phone=16137688175
&<公共请求参数>
```

输出示例

```
{
"Response": {
"Description": "信息不一致",
"Isp": "电信",
"RequestId": "5d3e13ad-ff97-409c-ad98-373158369b43",
"Result": "-4"
}
}
```

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.DecryptSystemError	解密失败。
FailedOperation.StsUnAuthErrError	STS未授权。
FailedOperation.UnKnown	内部未知错误。
InvalidParameter	参数错误。
InvalidParameter.UnsupportEncryptField	存在不加密的字段，请参考文档修改。
InvalidParameterValue	参数取值错误。
UnauthorizedOperation.Arrears	账号已欠费。
UnauthorizedOperation.ChargeStatusException	计费状态异常。
UnauthorizedOperation.NonAuthorize	账号未实名。
UnauthorizedOperation.Nonactivated	未开通服务。

手机号三要素核验（联通）

最近更新时间：2025-05-26 19:47:52

1. 接口描述

接口请求域名： faceid.tencentcloudapi.com 。

本接口用于校验中国联通手机号、姓名和身份证号的真实性和一致性。中国联通支持的手机号段详情请查阅运营商类文档。

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：PhoneVerificationCUCC。
Version	是	String	公共参数，本接口取值：2018-03-01。
Region	否	String	公共参数，本接口不需要传递此参数。
IdCard	是	String	身份证号。 示例值：11204416541220243X
Name	是	String	姓名。 示例值：韦小宝
Phone	是	String	手机号。 示例值：16137688175
Encryption	否	Encryption	敏感数据加密信息。 - 对传入信息（姓名、身份证号、手机号）有加密需求的用户可使用此参数，详情请点击左侧链接。

3. 输出参数

参数名称	类型	描述
Result	String	认证结果码，收费情况如下。 - 收费结果码： 0: 认证通过。 -4: 信息不一致（手机号已实名，但姓名和身份证号与实名信息不一致）。 - 不收费结果码： -6: 手机号码不合法。 -7: 身份证号码有误。 -8: 姓名校验不通过。 -9: 没有记录。 -11: 验证中心服务繁忙。 示例值：0
Isp	String	运营商名称。 - 取值范围为["移动","联通","电信",""]。 示例值：联通
Description	String	业务结果描述。 示例值：比对一致
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 联通手机号三要素核验一致示例

输入示例

```
https://faceid.tencentcloudapi.com/?Action=PhoneVerificationCUCC
&IdCard=11204416541220243X
&Name=韦小宝
&Phone=16137688175
&<公共请求参数>
```

输出示例

```
{
"Response": {
"Result": "0",
"Description": "认证通过",
"Isp": "联通",
"RequestId": "a5fdb909-5ee6-43ff-a152-bb1b9744ee8b"
}
}
```

示例2 联通手机号三要素核验不一致示例

输入示例

```
https://faceid.tencentcloudapi.com/?Action=PhoneVerificationCUCC
&IdCard=11204416541220243X
&Name=韦小宝
&Phone=16137688175
&<公共请求参数>
```

输出示例

```
{
"Response": {
"Description": "信息不一致",
"Isp": "联通",
"RequestId": "74182f34-cc90-40b6-b96d-e4189726f0ba",
"Result": "-4"
}
}
```

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.DecryptSystemError	解密失败。
FailedOperation.StsUnAuthErrError	STS未授权。
FailedOperation.UnKnown	内部未知错误。
InvalidParameter	参数错误。
InvalidParameter.UnsupportEncryptField	存在不加密的字段，请参考文档修改。
InvalidParameterValue	参数取值错误。
UnauthorizedOperation.Arrears	账号已欠费。
UnauthorizedOperation.ChargeStatusException	计费状态异常。
UnauthorizedOperation.NonAuthorize	账号未实名。
UnauthorizedOperation.Nonactivated	未开通服务。

手机号二要素核验

最近更新时间：2025-05-26 19:47:54

1. 接口描述

接口请求域名： faceid.tencentcloudapi.com 。

手机号二要素核验接口用于校验手机号和姓名的真实性和一致性，支持的手机号段详情请查阅运营商类文档。

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：CheckPhoneAndName。
Version	是	String	公共参数，本接口取值：2018-03-01。
Region	否	String	公共参数，本接口不需要传递此参数。
Mobile	是	String	机号。 示例值：16137688175
Name	是	String	姓名。 示例值：韦小宝
Encryption	否	Encryption	敏感数据加密信息。 - 对传入信息（姓名、手机号）有加密需求的用户可使用此参数，详情请点击左侧链接。

3. 输出参数

参数名称	类型	描述
Result	String	认证结果码，收费情况如下。 - 收费结果码： 0: 验证结果一致。 1: 验证结果不一致。 - 不收费结果码： -1:查无记录。 -2:引擎未知错误。 -3:引擎服务异常。 -4:姓名校验不通过。 -5:手机号码不合法。 -6: 认证次数超过当日限制，请次日重试。 示例值：0
Description	String	业务结果描述。 示例值：验证结果一致
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 手机号二要素核验一致示例

输入示例

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

{
"Mobile": "16137688175",
"Name": "韦小宝"
}
```

输出示例

```
{
"Response": {
"Result": "0",
"Description": "验证结果一致",
"RequestId": "368a9e91-71dc-49c7-b622-c4a300ed7370"
}
}
```

示例2 手机号二要素核验不一致示例

输入示例

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

{
"Mobile": "16137688175",
"Name": "韦小宝"
}
```

输出示例

```
{
"Response": {
"Description": "验证结果不一致",
"RequestId": "eb0e4b76-4617-46f5-8582-7cc58fd20115",
"Result": "1"
}
}
```

5. 开发者资源

腾讯云 API 平台

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

API Inspector

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

SDK

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

- Tencent Cloud SDK 3.0 for Python: GitHub, Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub, Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub, Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub, Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub, Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub, Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub, Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub, Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

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.DecryptSystemError	解密失败。
FailedOperation.StsUnAuthErrError	STS未授权。
FailedOperation.UnKnown	内部未知错误。
InternalError	内部错误。
InvalidParameter.UnsupportEncryptField	存在不加密的字段，请参考文档修改。
UnauthorizedOperation.Arrears	账号已欠费。
UnauthorizedOperation.ChargeStatusException	计费状态异常。
UnauthorizedOperation.NonAuthorize	账号未实名。
UnauthorizedOperation.Nonactivated	未开通服务。

人脸核身PaaS服务相关接口
获取动作顺序

最近更新时间：2025-05-26 19:47:48

1. 接口描述

接口请求域名： faceid.tencentcloudapi.com 。

使用动作活体检测模式前，需调用本接口获取动作顺序。

默认接口请求频率限制：100次/秒。

推荐使用 API Explorer　点击调试

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

2. 输入参数

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

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：GetActionSequence。
Version	是	String	公共参数，本接口取值：2018-03-01。
Region	否	String	公共参数，本接口不需要传递此参数。
ActionType	否	String	默认不需要使用。 示例值：0

3. 输出参数

参数名称	类型	描述
ActionSequence	String	动作顺序，例如：2,1 or 1,2。 - 1代表张嘴，2代表闭眼。 示例值：2,1
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 获取动作顺序成功示例

输入示例

```
https://faceid.tencentcloudapi.com/?Action=GetActionSequence
&<公共请求参数>
```

输出示例

```
{
"Response": {
"ActionSequence": "1,2",
"RequestId": "f904f4cf-75db-4f8f-a5ec-dc4f942c7f7a"
```

```
}
}
```

示例2 获取动作顺序失败示例

输入示例

```
https://faceid.tencentcloudapi.com/?Action=GetActionSequence
&<公共请求参数>
```

输出示例

```
{
"Response": {
"Error": {
"Code": "UnauthorizedOperation.Nonactivated",
"Message": "未开通服务。"
},
"RequestId": "4a75418f-96f7-4045-955f-32a7dd1b08bf"
}
}
```

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.UnKnown	内部未知错误。
InvalidParameter	参数错误。
InvalidParameterValue	参数取值错误。
UnauthorizedOperation	未授权操作。

错误码	描述
UnauthorizedOperation.Arrears	账号已欠费。
UnauthorizedOperation.NonAuthorize	账号未实名。
UnauthorizedOperation.Nonactivated	未开通服务。
UnsupportedOperation	操作不支持。

获取数字验证码

最近更新时间：2025-05-26 19:47:48

1. 接口描述

接口请求域名： faceid.tencentcloudapi.com 。

使用数字活体检测模式前，需调用本接口获取数字验证码。

默认接口请求频率限制：100次/秒。

推荐使用 API Explorer

点击调试

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

2. 输入参数

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

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：GetLiveCode。
Version	是	String	公共参数，本接口取值：2018-03-01。
Region	否	String	公共参数，本接口不需要传递此参数。

3. 输出参数

参数名称	类型	描述
LiveCode	String	数字验证码。 示例值：4392
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 获取数字验证码成功示例

输入示例

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

{}
```

输出示例

```
{
"Response": {
"LiveCode": "4392",
"RequestId": "f904f4cf-75db-4f8f-a5ec-dc4f942c7f7a"
```

```
}
}
```

示例2 获取数字验证码失败示例

输入示例

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

{}
```

输出示例

```
{
"Response": {
"Error": {
"Code": "UnauthorizedOperation.Nonactivated",
"Message": "未开通服务。"
},
"RequestId": "4a75418f-96f7-4045-955f-32a7dd1b08bf"
}
}
```

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.InvalidAuthorization	CAM签名/鉴权错误。
FailedOperation.UnKnown	内部未知错误。
InvalidParameter	参数错误。
InvalidParameterValue	参数取值错误。
UnauthorizedOperation	未授权操作。
UnauthorizedOperation.Arrears	账号已欠费。
UnauthorizedOperation.NonAuthorize	账号未实名。
UnauthorizedOperation.Nonactivated	未开通服务。
UnsupportedOperation	操作不支持。

活体人脸比对

最近更新时间：2025-05-26 19:47:57

1. 接口描述

接口请求域名： faceid.tencentcloudapi.com 。

传入视频和照片，先判断视频中是否为真人，判断为真人后，再判断该视频中的人与上传照片是否属于同一个人。

默认接口请求频率限制：100次/秒。

推荐使用 API Explorer　点击调试

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

2. 输入参数

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

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：LivenessCompare。
Version	是	String	公共参数，本接口取值：2018-03-01。
Region	否	String	公共参数，此参数为可选参数。
LivenessType	是	String	活体检测类型。 - 取值：LIP/ACTION/SILENT。 - LIP为数字模式，ACTION为动作模式，SILENT为静默模式，三种模式选择一种传入。 示例值：LIP
ImageBase64	否	String	用于人脸比对的照片的Base64值。 - Base64编码后的图片数据大小不超过3M，仅支持jpg、png格式。 - 请使用标准的Base64编码方式(带=补位)，编码规范参考RFC4648。 - 图片的 ImageUrl、ImageBase64 必须提供一个，如果都提供，只使用 ImageBase64。 示例值：/9j/4AAQSkZJRg.....s97n//2Q==
ImageUrl	否	String	用于人脸比对照片的URL地址。 - 图片下载后经Base64编码后的数据大小不超过3M，仅支持jpg、png格式。 - 图片的 ImageUrl、ImageBase64 必须提供一个，如果都提供，只使用 ImageBase64。 - 图片存储于腾讯云的 Url 可保障更高的下载速度和稳定性，建议图片存储于腾讯云。 - 非腾讯云存储的 Url 速度和稳定性可能受一定影响。 示例值：http://www.qq.com/image.jpg
ValidateData	否	String	验证数据。 - 数字模式传参：传数字验证码，验证码需先调用获取数字验证码接口得到； - 动作模式传参：传动作顺序，动作顺序需先调用获取动作顺序接口得到； - 静默模式传参：空。 示例值：2,1

Optional	否	String	额外配置，传入JSON字符串。
			
			- 格式如下：
			{
			"BestFrameNum": 2 //需要返回多张最佳截图，取值范围2-10
			}
			示例值：{"BestFrameNum":1}

VideoBase64	否	String	用于活体检测的视频，视频的Base64值。 - Base64编码后的大小不超过8M，支持mp4、avi、flv格式。 - 请使用标准的Base64编码方式(带=补位)，编码规范参考RFC4648。 - 视频的 VideoUrl、VideoBase64 必须提供一个，如果都提供，只使用 VideoBase64。 示例值：AAAAIGZ0eX...0tLS0tLS8=
VideoUrl	否	String	用于活体检测的视频Url 地址。 - 视频下载后经Base64编码后不超过 8M，视频下载耗时不超过4S，支持mp4、avi、flv格式。

参数名称	必选	类型	描述
			- 视频的 VideoUrl、VideoBase64 必须提供一个，如果都提供，只使用 VideoBase64。 - 建议视频存储于腾讯云的 Url 可保障更高的下载速度和稳定性，建议视频存储于腾讯云。 - 非腾讯云存储的 Url 速度和稳定性可能受一定影响。 示例值：http://www.qq.com/video.mp4

3. 输出参数

参数名称	类型	描述
BestFrameBase64	String	验证通过后的视频最佳截图照片。 - 照片为BASE64编码后的值，jpg格式。 注意：此字段可能返回 null，表示取不到有效值。 示例值：/9j/4AAQSkZJRg.....s97n//2Q==
Sim	Float	相似度。 - 取值范围 [0.00, 100.00]。 - 推荐相似度大于等于70时可判断为同一人，可根据具体场景自行调整阈值（阈值70的误通过率为千分之一，阈值80的误通过率是万分之一）。 示例值：88.33
Result	String	业务错误码。 - 成功情况返回Success。 - 错误情况请参考下方错误码，列表中FailedOperation部分。 示例值：Success
Description	String	业务结果描述。 示例值：成功
BestFrameList	Array of String	最佳截图列表。 - 仅在配置了返回多张最佳截图时返回。 注意：此字段可能返回 null，表示取不到有效值。 示例值：["/9j/4AAQSk...vv56s//9k="]
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 静默活体人脸比对成功示例

输入示例

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

{
"LivenessType": "SILENT",
"ImageBase64": "/9j/4AAQSk...GArFYH/9k=",
"VideoBase64": "AAAAIGZ0eX...cBQCKAowc="
}
```

输出示例

```
{
"Response": {
"Result": "Success",
"Description": "成功",
```

```
"BestFrameBase64": "/9j/4AAQSk...W8+nU//9k=",
"BestFrameList": [
"/9j/4AAQSk...px72+8/9k="
],
"Sim": 89.88,
"RequestId": "f904f4cf-75db-4f8f-a5ec-dc4f942c7f7a"
}
}
```


示例2 静默活体人脸比对失败示例

输入示例

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

{
"LivenessType": "SILENT",
"ImageBase64": "/9j/4AAQSk...GArFYH/9k=",
"VideoBase64": "AAAAIGZ0eX...cBQCKAowc="
}
```

输出示例


```
{
"Response": {
"BestFrameBase64": "/9j/4AAQSk...RgkelAH//Z",
"BestFrameList": [],
"Description": "实人检测失败",
"RequestId": "09fe2045-af63-43d5-8d7a-d9f7834ca62e",
"Result": "FailedOperation.SilentDetectFail",
"Sim": 0
}
}
```


5. 开发者资源

腾讯云 API 平台

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

API Inspector

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

SDK

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

- Tencent Cloud SDK 3.0 for Python: GitHub, Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub, Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub, Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub, Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub, Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub, Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub, Gitee

- Tencent Cloud SDK 3.0 for Ruby: GitHub, Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
AuthFailure.InvalidAuthorization	CAM签名/鉴权错误。
FailedOperation.ActionCloseEye	未检测到闭眼动作。
FailedOperation.ActionFaceClose	脸离屏幕太近。
FailedOperation.ActionFaceFar	脸离屏幕太远。
FailedOperation.ActionFaceLeft	脸离屏幕太左。
FailedOperation.ActionFaceRight	脸离屏幕太右。
FailedOperation.ActionFirstAction	未检测到动作配合。
FailedOperation.ActionLightDark	光线太暗。
FailedOperation.ActionLightStrong	光线太强。
FailedOperation.ActionNodetectFace	未能检测到完整人脸。
FailedOperation.ActionOpenMouth	未检测到张嘴动作。
FailedOperation.CompareFail	比对失败。
FailedOperation.CompareLowSimilarity	比对相似度未达到通过标准。
FailedOperation.CompareSystemError	调用比对引擎接口出错。
FailedOperation.CompressVideoError	The video compression failed. Please try again or reduce the size of the input video.
FailedOperation.DownLoadError	文件下载失败。
FailedOperation.DownLoadTimeoutError	文件下载超时。
FailedOperation.LifePhotoDetectFaces	检测到多张人脸。
FailedOperation.LifePhotoDetectFake	实人比对没通过。
FailedOperation.LifePhotoDetectNoFaces	未能检测到完整人脸。
FailedOperation.LifePhotoPoorQuality	传入图片分辨率太低，请重新上传。
FailedOperation.LifePhotoSizeError	传入图片过大或过小。
FailedOperation.LipFaceIncomplete	脸部未完整露出。
FailedOperation.LipMoveSmall	嘴唇动作幅度过小。
FailedOperation.LipNetFailed	视频拉取失败，请重试。
FailedOperation.LipSizeError	视频为空，或大小不合适，请控制录制时长在6s左右。
FailedOperation.LipVideoInvalid	视频格式有误。
FailedOperation.LipVideoQuaility	视频像素太低。
FailedOperation.LipVoiceDetect	未检测到声音。
FailedOperation.LipVoiceLow	视频声音太小。
FailedOperation.LipVoiceRecognize	声音识别失败。

错误码	描述
FailedOperation.LivessBestFrameError	人脸检测失败，无法提取比对照。
FailedOperation.LivessDetectFail	活体检测没通过。
FailedOperation.LivessDetectFake	疑似非真人录制。
FailedOperation.LivessSystemError	调用活体引擎接口出错。
FailedOperation.LivessUnknownError	视频实人检测没通过。
FailedOperation.SilentDetectFail	实人检测失败。
FailedOperation.SilentEyeLiveFail	眼睛检测失败。
FailedOperation.SilentFaceDetectFail	视频未检测到人脸。
FailedOperation.SilentFaceQualityFail	视频中人脸质量低。
FailedOperation.SilentFaceWithMaskFail	检测到带口罩。
FailedOperation.SilentMouthLiveFail	嘴巴检测失败。
FailedOperation.SilentMultiFaceFail	视频检测中有多个人脸。
FailedOperation.SilentPictureLiveFail	疑似翻拍。
FailedOperation.SilentThreshold	实人检测未达到通过标准。
FailedOperation.SilentTooShort	视频录制时间过短，请录制2秒以上的视频。
FailedOperation.UnKnown	内部未知错误。
InvalidParameter	参数错误。
InvalidParameterValue	参数取值错误。
UnauthorizedOperation	未授权操作。
UnauthorizedOperation.Arrears	账号已欠费。
UnauthorizedOperation.NonAuthorize	账号未实名。
UnauthorizedOperation.Nonactivated	未开通服务。
UnsupportedOperation	操作不支持。

活体人脸核身

最近更新时间：2025-05-26 19:47:56

1. 接口描述

接口请求域名： faceid.tencentcloudapi.com 。

传入视频和身份信息，先判断视频中是否为真人，判断为真人后，再判断该视频中的人与权威库的证件照是否属于同一个人。

默认接口请求频率限制：100次/秒。

推荐使用 API Explorer　点击调试

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

2. 输入参数

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

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：LivenessRecognition。
Version	是	String	公共参数，本接口取值：2018-03-01。
Region	否	String	公共参数，此参数为可选参数。
IdCard	是	String	身份证号。 示例值：11204416541220243X
Name	是	String	姓名。 - 中文请使用UTF-8编码。 示例值：韦小宝
LivenessType	是	String	活体检测类型。 - 取值：LIP/ACTION/SILENT。 - LIP为数字模式，ACTION为动作模式，SILENT为静默模式，三种模式选择一种传入。 示例值：SILENT
VideoBase64	否	String	用于活体检测的视频，视频的BASE64值； BASE64编码后的大小不超过8M，支持mp4、avi、flv格式。 示例值：AAAAIGZ0eX...0tLS0tLS8=
VideoUrl	否	String	用于活体检测的视频Url 地址。 - 视频下载后经Base64编码不超过 8M，视频下载耗时不超过4S，支持mp4、avi、flv格式。 - 视频的 VideoUrl、VideoBase64 必须提供一个，如果都提供，只使用 VideoBase64。 - 建议视频存储于腾讯云的 Url 可保障更高的下载速度和稳定性，建议视频存储于腾讯云。 - 非腾讯云存储的 Url 速度和稳定性可能受一定影响。 示例值：http://www.qq.com/image.jpg
ValidateData	否	String	验证数据。 - 数字模式传参：传数字验证码，验证码需先调用获取数字验证码接口得到； - 动作模式传参：传动作顺序，动作顺序需先调用获取动作顺序接口得到； - 静默模式传参：空。 示例值：4903

Optional	否	String	额外配置，传入JSON字符串。
			- 格式如下：
			{
			"BestFrameNum": 2 //需要返回多张最佳截图，取值范围2-10
			}
			示例值：{"BestFrameNum":1}

Encryption	否	Encryption	敏感数据加密信息。 - 对传入信息（姓名、身份证号）有加密需求的用户可使用此参数，详情请点击左侧链接。

3. 输出参数

参数名称	类型	描述
BestFrameBase64	String	验证通过后的视频最佳截图照片。 - 照片为BASE64编码后的值，jpg格式。 注意：此字段可能返回 null，表示取不到有效值。 示例值：/9j/4AAQSkZJRg.....s97n//2Q==
Sim	Float	相似度。 - 取值范围 [0.00, 100.00]。 - 推荐相似度大于等于70时可判断为同一人，可根据具体场景自行调整阈值（阈值70的误通过率为千分之一，阈值80的误通过率是万分之一） 示例值：89.88
Result	String	业务错误码。 - 成功情况返回Success。 - 错误情况请参考下方错误码 列表中FailedOperation部分 示例值：Success
Description	String	业务结果描述。 示例值：成功
BestFrameList	Array of String	最佳截图列表，仅在配置了返回多张最佳截图时返回。 注意：此字段可能返回 null，表示取不到有效值。 示例值：["/9j/4AAQSk...Kff7/+Af/Z"]
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 静默活体人脸核身成功示例

输入示例

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

{
"IdCard": "11204416541220243X",
"LivenessType": "SILENT",
"Name": "韦小宝",
"VideoBase64": "AAAAIGZ0eX...0tLS0tLS8="
}
```

输出示例

```
{
"Response": {
"Result": "Success",
"Description": "成功",
"BestFrameBase64": "/9j/4AAQSkZJRg.....s97n//2Q==",
"BestFrameList": [
"/9j/4AAQSk...ZpGq7an//Z"
],
"Sim": 89.88,
"RequestId": "f904f4cf-75db-4f8f-a5ec-dc4f942c7f7a"
```

```
}
}
```

示例2 数字活体人脸核身失败示例

输入示例

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

{
"IdCard": "11204416541220243X",
"LivenessType": "LIP",
"Name": "韦小宝",
"VideoBase64": "AAAAIGZ0eX...0tLS0tLS8=",
"ValidateData": "4903"
}
```

输出示例

```
{
"Response": {
"BestFrameBase64": "/9j/4AAQSk...ZpGq7an//Z",
"BestFrameList": [],
"Description": "声音识别失败",
"RequestId": "0187d319-86ac-4687-807f-7733d54ba0c0",
"Result": "FailedOperation.LipVoiceRecognize",
"Sim": 0
}
}
```

5. 开发者资源

腾讯云 API 平台

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

API Inspector

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

SDK

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

- Tencent Cloud SDK 3.0 for Python: GitHub, Gitee
- Tencent Cloud SDK 3.0 for Java: GitHub, Gitee
- Tencent Cloud SDK 3.0 for PHP: GitHub, Gitee
- Tencent Cloud SDK 3.0 for Go: GitHub, Gitee
- Tencent Cloud SDK 3.0 for Node.js: GitHub, Gitee
- Tencent Cloud SDK 3.0 for .NET: GitHub, Gitee
- Tencent Cloud SDK 3.0 for C++: GitHub, Gitee
- Tencent Cloud SDK 3.0 for Ruby: GitHub, Gitee

命令行工具

- Tencent Cloud CLI 3.0

6. 错误码

以下仅列出了接口业务逻辑相关的错误码，其他错误码详见 公共错误码。

错误码	描述
AuthFailure.InvalidAuthorization	CAM签名/鉴权错误。
FailedOperation.ActionCloseEye	未检测到闭眼动作。
FailedOperation.ActionFaceClose	脸离屏幕太近。
FailedOperation.ActionFaceFar	脸离屏幕太远。
FailedOperation.ActionFaceLeft	脸离屏幕太左。
FailedOperation.ActionFaceRight	脸离屏幕太右。
FailedOperation.ActionFirstAction	未检测到动作配合。
FailedOperation.ActionLightDark	光线太暗。
FailedOperation.ActionLightStrong	光线太强。
FailedOperation.ActionNodetectFace	未能检测到完整人脸。
FailedOperation.ActionOpenMouth	未检测到张嘴动作。
FailedOperation.CompareFail	比对失败。
FailedOperation.CompareLibServiceUnavailable	比对库源维护中，暂时不可用
FailedOperation.CompareLowSimilarity	比对相似度未达到通过标准。
FailedOperation.CompareSystemError	调用比对引擎接口出错。
FailedOperation.DecryptSystemError	解密失败。
FailedOperation.DownLoadError	文件下载失败。
FailedOperation.DownLoadTimeoutError	文件下载超时。
FailedOperation.IdFormatError	输入的身份证号有误。
FailedOperation.IdNameMisMatch	姓名和身份证号不一致，请核实后重试。
FailedOperation.IdNoExistSystem	库中无此号，请到户籍所在地进行核实。
FailedOperation.IdPhotoNoExist	库中无此号照片，请到户籍所在地进行核实。
FailedOperation.IdPhotoPoorQuality	证件图片质量差，请更新后重试。
FailedOperation.IdentityAuthLimitExceeded	姓名/身份证号认证次数超过当日限制，请次日重试
FailedOperation.LifePhotoDetectFaces	检测到多张人脸。
FailedOperation.LifePhotoDetectFake	实人比对没通过。
FailedOperation.LifePhotoDetectNoFaces	未能检测到完整人脸。
FailedOperation.LifePhotoSizeError	传入图片过大或过小。
FailedOperation.LipFaceIncomplete	脸部未完整露出。
FailedOperation.LipMoveSmall	嘴唇动作幅度过小。
FailedOperation.LipNetFailed	视频拉取失败，请重试。
FailedOperation.LipSizeError	视频为空，或大小不合适，请控制录制时长在6s左右。
FailedOperation.LipVideoInvalid	视频格式有误。

错误码	描述
FailedOperation.LipVideoQuaility	视频像素太低。
FailedOperation.LipVoiceDetect	未检测到声音。
FailedOperation.LipVoiceLow	视频声音太小。
FailedOperation.LipVoiceRecognize	声音识别失败。
FailedOperation.LivessBestFrameError	人脸检测失败，无法提取比对照。
FailedOperation.LivessDetectFail	活体检测没通过。
FailedOperation.LivessDetectFake	疑似非真人录制。
FailedOperation.LivessSystemError	调用活体引擎接口出错。
FailedOperation.LivessUnknownError	视频实人检测没通过。
FailedOperation.NameFormatError	输入的姓名有误。
FailedOperation.SilentDetectFail	实人检测失败。
FailedOperation.SilentThreshold	实人检测未达到通过标准。
FailedOperation.SilentTooShort	视频录制时间过短，请录制2秒以上的视频。
FailedOperation.StsUnAuthErrError	STS未授权。
FailedOperation.UnKnown	内部未知错误。
InvalidParameter	参数错误。
InvalidParameter.UnsupportEncryptField	存在不加密的字段，请参考文档修改。
InvalidParameterValue	参数取值错误。
UnauthorizedOperation	未授权操作。
UnauthorizedOperation.Arrears	账号已欠费。
UnauthorizedOperation.NonAuthorize	账号未实名。
UnauthorizedOperation.Nonactivated	未开通服务。
UnsupportedOperation	操作不支持。

照片人脸核身(V2.0)

最近更新时间：2025-05-26 19:47:47

1. 接口描述

接口请求域名： faceid.tencentcloudapi.com 。

传入照片和身份信息，判断该照片与权威库的证件照是否属于同一个人。

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：ImageRecognitionV2。
Version	是	String	公共参数，本接口取值：2018-03-01。
Region	否	String	公共参数，本接口不需要传递此参数。
IdCard	是	String	身份证号。 示例值：11204416541220243X
Name	是	String	姓名。 - 中文请使用UTF-8编码。 示例值：韦小宝
ImageBase64	是	String	用于人脸比对的照片，图片的Base64值； Base64编码后的图片数据大小不超过3M，仅支持jpg、png格式。 请使用标准的Base64编码方式(带=补位)，编码规范参考RFC4648。 示例值：/9j/4AAQSkZJRg.....s97n//2Q==
Optional	否	String	本接口不需要传递此参数。 示例值：92424b
Encryption	否	Encryption	敏感数据加密信息。 - 对传入信息（姓名、身份证号）有加密需求的用户可使用此参数，详情请点击左侧链接。
Extra	否	String	自定义描述字段。 - 用于描述调用业务信息，出参中将返回此描述字段。 - 每个自定义描述字段支持[1,10]个字符。 示例值：开户业务

3. 输出参数

参数名称	类型	描述
Sim	Float	相似度。 - 取值范围 [0.00, 100.00]。 - 推荐相似度大于等于70时可判断为同一人，可根据具体场景自行调整阈值（阈值70的误通过率为千分之一，阈值80的误通过率是万分之一） 示例值：89.88
Result	String	业务错误码。 - 成功情况返回Success。 - 错误情况请参考下方错误码 列表中FailedOperation部分 示例值：Success

参数名称	类型	描述
Description	String	业务结果描述。 示例值：成功
Extra	String	调用接口中自定义的描述字段。 示例值：开户业务
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 照片人脸核身成功示例

输入示例

```
https://faceid.tencentcloudapi.com/?Action=ImageRecognitionV2
&IdCard=11204416541220243X
&Name=韦小宝
&ImageBase64=/9j/4AAQSkZJRg.....s97n//2Q==
&<公共请求参数>
```

输出示例

```
{
"Response": {
"Result": "Success",
"Description": "成功",
"Sim": 89.88,
"RequestId": "f904f4cf-75db-4f8f-a5ec-dc4f942c7f7a"
}
}
```

示例2 照片人脸核身比对相似度未达标示例

输入示例

```
https://faceid.tencentcloudapi.com/?Action=ImageRecognitionV2
&IdCard=11204416541220243X
&Name=韦小宝
&ImageBase64=/9j/4AAQSkZJRg.....s97n//2Q==
&<公共请求参数>
```

输出示例

```
{
"Response": {
"Description": "比对相似度未达到通过标准",
"RequestId": "e9e198e4-4fa8-49f0-a67e-f8053bc49201",
"Result": "FailedOperation.CompareLowSimilarity",
"Sim": 26.04
}
}
```

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.CompareFail	比对失败。
FailedOperation.CompareLibServiceUnavailable	比对库源维护中，暂时不可用
FailedOperation.CompareLowSimilarity	比对相似度未达到通过标准。
FailedOperation.CompareSystemError	调用比对引擎接口出错。
FailedOperation.DecryptSystemError	解密失败。
FailedOperation.IdFormatError	输入的身份证号有误。
FailedOperation.IdNameMisMatch	姓名和身份证号不一致，请核实后重试。
FailedOperation.IdNoExistSystem	库中无此号，请到户籍所在地进行核实。
FailedOperation.IdPhotoNoExist	库中无此号照片，请到户籍所在地进行核实。
FailedOperation.IdPhotoPoorQuality	证件图片质量差，请更新后重试。
FailedOperation.IdPhotoSystemNoanswer	客户库自建库或认证中心返照失败，请稍后再试。
FailedOperation.IdentityAuthLimitExceeded	姓名/身份证号认证次数超过当日限制，请次日重试
FailedOperation.LifePhotoDetectFaces	检测到多张人脸。
FailedOperation.LifePhotoDetectFake	实人比对没通过。
FailedOperation.LifePhotoDetectNoFaces	未能检测到完整人脸。
FailedOperation.LifePhotoPoorQuality	传入图片分辨率太低，请重新上传。
FailedOperation.LifePhotoSizeError	传入图片过大或过小。
FailedOperation.NameFormatError	输入的姓名有误。
FailedOperation.StsUnAuthErrError	STS未授权。
FailedOperation.UnKnown	内部未知错误。
InvalidParameter	参数错误。

错误码	描述
InvalidParameter.UnsupportEncryptField	存在不加密的字段，请参考文档修改。
InvalidParameterValue	参数取值错误。
UnauthorizedOperation	未授权操作。
UnauthorizedOperation.Arrears	账号已欠费。
UnauthorizedOperation.NonAuthorize	账号未实名。
UnauthorizedOperation.Nonactivated	未开通服务。
UnsupportedOperation	操作不支持。

身份证人像照片验真

最近更新时间：2025-05-26 19:47:48

1. 接口描述

接口请求域名： faceid.tencentcloudapi.com 。

传入身份证人像面照片，识别身份证照片上的信息，并将姓名、身份证号、身份证人像照片与权威库的证件照进行比对，是否属于同一个人，从而验证身份证信息的真实性。

默认接口请求频率限制：100次/秒。

推荐使用 API Explorer　点击调试

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

2. 输入参数

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

参数名称	必选	类型	描述
Action	是	String	公共参数，本接口取值：CheckIdCardInformation。
Version	是	String	公共参数，本接口取值：2018-03-01。
Region	否	String	公共参数，本接口不需要传递此参数。
ImageBase64	否	String	身份证人像面的 Base64 值。 - 支持的图片格式：PNG、JPG、JPEG，暂不支持 GIF 格式。 - 支持的图片大小：所下载图片经Base64编码后不超过 7M。 - 请使用标准的Base64编码方式(带=补位)，编码规范参考RFC4648。 - ImageBase64、ImageUrl二者必须提供其中之一。若都提供了，则按照ImageUrl>ImageBase64的优先级使用参数。 示例值：/9j/4AAQSkZJRg.....s97n//2Q==
ImageUrl	否	String	身份证人像面的 Url 地址 - 支持的图片格式：PNG、JPG、JPEG，暂不支持 GIF 格式。 - 支持的图片大小：所下载图片经 Base64 编码后不超过 3M。图片下载时间不超过 3 秒。 - 图片存储于腾讯云的 Url 可保障更高的下载速度和稳定性，建议图片存储于腾讯云。 - 非腾讯云存储的 Url 速度和稳定性可能受一定影响。 示例值：http://www.qq.com/image.jpg
Config	否	String	配置。 - 以下可选字段均为bool 类型，默认false。 CopyWarn，复印件告警。 BorderCheckWarn，边框和框内遮挡告警。 ReshootWarn，翻拍告警。 DetectPsWarn，PS检测告警（疑似存在PS痕迹）。 TempIdWarn，临时身份证告警。 Quality，图片质量告警（评价图片模糊程度）。 - SDK 设置方式参考： Config = Json.stringify({"CopyWarn":true,"ReshootWarn":true})。 - API 3.0 Explorer 设置方式参考： Config = {"CopyWarn":true,"ReshootWarn":true}。 示例值： {"CopyWarn":true,"ReshootWarn":true}
IsEncrypt	否	Boolean	是否需要对返回中的敏感信息进行加密。 - 默认false。 - 敏感信息包括：Response.IdNum、Response.Name。 示例值：false
IsEncryptResponse	否	Boolean	是否需要对响应体加密。 示例值：false

参数名称	必选	类型	描述
Encryption	否	Encryption	是否需要对返回中的敏感信息进行加密,需指定加密算法Algorithm、CBC加密的初始向量、加密后的对称密钥。

3. 输出参数

参数名称	类型	描述
Sim	Float	相似度。 - 取值范围 [0.00, 100.00]。 - 推荐相似度大于等于70时可判断为同一人，可根据具体场景自行调整阈值（阈值70的误通过率为千分之一，阈值80的误通过率是万分之一）。 示例值：99.76
Result	String	业务错误码。- 成功情况返回Success。- 错误情况请参考下方错误码 列表中FailedOperation部分 示例值：FailedOperation.OcrWarningOccurred
Description	String	业务结果描述。 示例值：成功
Name	String	姓名。 示例值：韦小宝
Sex	String	性别。 示例值：男
Nation	String	民族。 示例值：汉
Birth	String	出生日期。 示例值：1654/12/20
Address	String	地址。 示例值：北京市东城区景山前街4号紫禁城敬事房
IdNum	String	身份证号。 示例值：11204416541220243X
Portrait	String	身份证头像照片的base64编码，如果抠图失败会拿整张身份证做比对并返回空。 示例值：/9j/4AAQSkZJRg.....s97n//2Q==
Warnings	String	告警信息。 - 当在Config中配置了告警信息会停止人像比对，Result返回错误（FailedOperation.OcrWarningOccurred）并有此告警信息。 - Code 告警码列表和释义： '-9101'：身份证边框不完整告警。 '-9102'：身份证复印件告警。 '-9103'：身份证翻拍告警。 '-9105'：身份证框内遮挡告警。 '-9104'：临时身份证告警。 '-9106'：身份证 PS 告警（疑似存在PS痕迹）。 '-8001'：图片模糊告警。 - 多个会用“|” 隔开，如 "-9101|-9106|-9104"。 示例值：-9101|-9106|-9104
Quality	Float	图片质量分数。 - 当请求Config中配置图片模糊告警该参数才有意义。 - 取值范围（0~100），目前默认阈值是50分，低于50分会触发模糊告警。 示例值：0
Encryption	Encryption	敏感数据加密信息。 注意：此字段可能返回 null，表示取不到有效值。
EncryptedBody	String	加密后的数据。 示例值：7a548b658cbe
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得 RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 身份证人像照片验真成功示例

输入示例

```
https://faceid.tencentcloudapi.com/?Action=CheckIdCardInformation
&ImageBase64=/9j/4AAQSkZJRg.....s97n//2Q==
&Config={"CopyWarn":true}
&<公共请求参数>
```

输出示例

```
{
"Response": {
"Address": "北京市东城区景山前街4号紫禁城敬事房",
"Birth": "1654/12/20",
"Description": "成功",
"Encryption": null,
"IdNum": "11204416541220243X",
"Name": "韦小宝",
"Nation": "汉",
"Portrait": "/9j/4AAQSk...m3Zt62P//Z",
"Quality": 98,
"RequestId": "8dc2b640-caad-4a13-ad8c-204d46d29fe1",
"Result": "Success",
"Sex": "男",
"Sim": 99.76,
"Warnings": ""
}
}
```

示例2 身份证人像照片验真失败示例

输入示例

```
https://faceid.tencentcloudapi.com/?Action=CheckIdCardInformation
&ImageBase64=/9j/4AAQSkZJRg.....s97n//2Q==
&Config={"CopyWarn":true}
&<公共请求参数>
```

输出示例

```
{
"Response": {
"Address": "北京市东城区景山前街4号紫禁城敬事房",
"Birth": "1654/12/20",
"Description": "出现OCR告警",
"IdNum": "11204416541220243X",
"Name": "韦小宝",
"Nation": "满",
"Portrait": "/9j/4AAQSk...Se/wDkf//Z",
"Quality": 99,
"Encryption": null,
"RequestId": "6e6cfaad-3271-4eb1-a203-2866100eb283",
"Result": "FailedOperation.OcrWarningOccurred",
```

```
"Sex": "男",
"Sim": 0,
"Warnings": "-9101"
}
}
```


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.CompareFail	比对失败。
FailedOperation.CompareLowSimilarity	比对相似度未达到通过标准。
FailedOperation.CompareSystemError	调用比对引擎接口出错。
FailedOperation.DownLoadError	文件下载失败。
FailedOperation.EmptyImageError	图片内容为空。
FailedOperation.EncryptSystemError	加密失败。
FailedOperation.FileSaveError	文件存储失败，请稍后重试。
FailedOperation.IdFormatError	输入的身份证号有误。
FailedOperation.IdNameMisMatch	姓名和身份证号不一致，请核实后重试。
FailedOperation.IdNoExistSystem	库中无此号，请到户籍所在地进行核实。
FailedOperation.IdPhotoNoExist	库中无此号照片，请到户籍所在地进行核实。
FailedOperation.IdPhotoPoorQuality	证件图片质量差，请更新后重试。
FailedOperation.IdPhotoSystemNoanswer	客户库自建库或认证中心返照失败，请稍后再试。
FailedOperation.ImageBlur	图片模糊。
FailedOperation.ImageDecodeFailed	图片解码失败。

错误码	描述
FailedOperation.ImageNoIdCard	图片中未检测到身份证。
FailedOperation.ImageSizeTooLarge	图片尺寸过大。
FailedOperation.LifePhotoDetectFaces	检测到多张人脸。
FailedOperation.LifePhotoDetectFake	实人比对没通过。
FailedOperation.LifePhotoDetectNoFaces	未能检测到完整人脸。
FailedOperation.LifePhotoPoorQuality	传入图片分辨率太低，请重新上传。
FailedOperation.LifePhotoSizeError	传入图片过大或过小。
FailedOperation.OcrFailed	Ocr识别失败。
FailedOperation.RequestLimitExceeded	调用次数超出限制。
FailedOperation.StsUnAuthErrError	STS未授权。
FailedOperation.UnKnown	内部未知错误。
InternalError.EncryptSystemError	加密失败。
InternalError.UnKnown	内部未知错误。
InvalidParameter	参数错误。
UnauthorizedOperation.Arrears	账号已欠费。
UnauthorizedOperation.ChargeStatusException	计费状态异常。
UnauthorizedOperation.Nonactivated	未开通服务。

AI人脸防护盾相关接口
AI人脸防护盾

最近更新时间：2025-05-26 19:47:48

1. 接口描述

接口请求域名： faceid.tencentcloudapi.com 。

基于多模态的AI大模型算法，提供对人脸图片、视频的防攻击检测能力，可针对性有效识别高仿真的AIGC换脸、高清翻拍、批量黑产攻击、水印等攻击痕迹，增强对图片和视频的防伪安全能力。

默认接口请求频率限制：5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：DetectAIFakeFaces。
Version	是	String	公共参数，本接口取值：2018-03-01。
Region	否	String	公共参数，此参数为可选参数。
FaceInput	否	String	传入需要进行检测的带有人脸的图片或视频（当前仅支持单人脸检测），使用base64编码的形式。 - 图片的Base64值： 建议整体图像480x640的分辨率，脸部 大小 100X100 以上。 Base64编码后的图片数据大小建议不超过3M、最大不可超过10M，仅支持jpg、png格式。 请使用标准的Base64编码方式(带=补位)，编码规范参考RFC4648。 - 视频的Base64值： Base64编码后的大小建议不超过8M、最大不可超过10M，支持mp4、avi、flv格式。 请使用标准的Base64编码方式(带=补位)，编码规范参考RFC4648。 视频时长最大支持20s，建议时长2～5s。 建议视频分辨率为480x640，帧率在25fps~30fps之间。 示例值：/9j/4AAQSkZJRg.....s97n//2Q== 示例值：/9j/4AAQSkZJRg.....s97n//2Q==
FaceInputType	否	Integer	传入的类型。 - 取值范围： 1：传入的是图片类型。 2：传入的是视频类型。 其他：返回错误码InvalidParameter。 示例值：1
Encryption	否	Encryption	是否需要对请求信息进行全包体加密。 - 支持的加密算法:AES-256-CBC、SM4-GCM。 - 有加密需求的用户可使用此参数，详情请点击左侧链接。
EncryptedBody	否	String	加密后的密文。 - 加密前的数据格式如下:{"FaceInput":"AAAAA","FaceInputType":1}。 示例值：fH6Fc1xHm.....UrjSoYE

3. 输出参数

参数名称	类型	描述
AttackRiskLevel	String	检测到的图片是否存在攻击。 - Low：无攻击风险。 - Mid：中度疑似攻击。 - High：高度疑似攻击。 示例值：Mid
AttackRiskDetailList	Array of AttackRiskDetail	检测到疑似的攻击痕迹列表。 - 说明：未检测到攻击痕迹时，返回空数组。 - 此出参仅作为结果判断的参考，实际应用仍建议使用AttackRiskLevel的结果。 示例值：[{ Type:"SuspectedSpoofingAttack" }]
ExtraInfo	ExtraInfo	额外信息。
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 不加密请求示例

输入示例

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

{
"FaceInputType": 1,
"FaceInput": "VBORw0KGgoAAAANSUhEUgAAAkYAAAI9CAYAAADfOLduAAABQ2lDQ1BJQ0MgUHJvZmlsZQAAKJFjYGASSCwoyGFhYGDIzSspCnJ3UoiIjFJgf
87AziDBwMcgwqCQmFxc4BgQ4ANUwgCjUcG3awyMIPqyLsgsz5I1BWZcn/xff7Y4dPvimWmY6lEAV0pqcTKQ/gPEyckFRSUMDIwJQLZyeUkBiN0CZIsUAR0FZM8
AsdMh7DUgdhKEfQCsJiTIGci+AmQLJGckpgDZT4BsnSQk8XQkNtReEOD0CFBwNTI3LiTgVpJBSWpFCYh2zi+oLMpMzyhRcASGUKqCZ16yno6CkYGRMQMDKLwhq
j/fAIcjoxgHQqzwKgODhTyQ8RQhlniBgWH3OgYG4Z8IMWUDBgYeoGn7/AsSixLhDmD8xlKcZmwEYXNvZ2Bgnfb//+dwBgZ2TQaGv9f///+9/f//v8sYGJhvMTA
c+AYADqxgzYM88zoAAAA4ZVhJZk1NACoAAAAIAAGHaQAEAAAAAQAAABoAAAAAAAKgAgAEAAAAAQAAAkagAwAEAAAAAQAAAj0AAAAi/M5dAAAQABJREFUeAHsv
Wd35EiSbYsIRlCkViW6pmq6Wry71vz/fzEf3n2fpufd"
}
```

输出示例

```
{
"Response": {
"AttackRiskDetailList": [
{
"Type": "SuspectedWatermark"
}
],
"AttackRiskLevel": "High",
"RequestId": "2dd93e6f-5121-4bac-8c64-d6ad646663d2"
}
}
```

示例2 异常示例

输入示例

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

{
"FaceInputType": 2,
"FaceInput": "AAAAHGZ0eXBtcDQyAAAAAWlzb21tcDQxbXA0MgAADXNtb292AAAAbG12aGQAAAAA2qNvttqjb7cAAB9AAACYFgABAAABAAAAAAAAAAAAAAAAAAA
AQAAAAAAAAAAAAAAAAAAAAAEAAAAAAAAAAAAAAAAAAEAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAADAAAJ/XRyYWsAAABcdGtoZAAAAAHao2+22qNvtw
AAAAEAAAAAAAACYFgAAAAAAAAAAAAAAAAAAAAAAAAAQAAAAAAAAAAAAAAAAAAAAEAAAAAAAAAAAAAAAAAAAEAAAAADwAAAAtAAAAAAAACRlZHRzAAAAHGVsc3QAAAAA
AAAAAQAAmBYAAAAUAAEAAAAAACXVtZGlhAAAAIG1kaGQAAAAA2qNvttqjb7cAAAJYAAALaFXEAAAAAAAxaGRscgAAAAAAAAAAdmlkZQAAAAAAAAAAAAAAAENvcm
UgTWVkaWEgVmlkZW8AAAAJHG1pbmYAAAAUdm1oZAAAAAEAAAAAAAAAAAAAACRkaW5mAAAAHGRyZWYAAAAAAAAAAAQ"
}
```

输出示例

```
{
"Response": {
"Error": {
"Code": "FailedOperation.ImageDecodeFailed",
"Message": "图片解码失败。"
},
"RequestId": "0e77ad29-ad65-4901-9efc-b49a6e0a357b"
}
}
```

示例3 全包体加密请求示例

输入示例

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

{
"Encryption": {
"Algorithm": "AES-256-CBC",
"EncryptList": [
"EncryptionBody"
],
"CiphertextBlob": "qwRTgn+52FAvnfrGL1oPBsXG4FLBd42YRw",
"Iv": "vqfsaEOpGJESA=="
},
"EncryptedBody": "v32onREKL6bk6WHOGFEjE2M6kvqybGCVTjlm"
}
```

输出示例

```
{
"Response": {
"AttackRiskDetailList": [
{
```

```
"Type": "SuspectedWatermark"
}
],
"AttackRiskLevel": "High",
"RequestId": "2dd93e6f-5121-4bac-8c64-d6ad646663d2"
}
}
```


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.CoveredFace	图中人脸存在遮挡，请传入无遮挡人脸图片
FailedOperation.DetectEngineSystemError	服务引擎调用失败，请重试
FailedOperation.ImageDecodeFailed	图片解码失败。
FailedOperation.IncompleteFace	未检测到完整人脸，请传入完整人脸图片
FailedOperation.PoorImageQuality	图片质量过差，请检查图片质量
FailedOperation.UnKnown	内部未知错误。
FailedOperation.VideoDecodeFailed	视频解码异常
FailedOperation.VideoDurationExceeded	视频时长过长，当前接口最大支持的视频时长为20s。
InvalidParameter	参数错误。
InvalidParameterValue	参数取值错误。
UnauthorizedOperation.Arrears	账号已欠费。
UnauthorizedOperation.ChargeStatusException	计费状态异常。
UnauthorizedOperation.Nonactivated	未开通服务。

其他接口
查询账单明细（微信渠道）

最近更新时间：2025-05-26 19:47:55

1. 接口描述

接口请求域名： faceid.tencentcloudapi.com 。

查询微信渠道服务（微信小程序、微信原生H5、微信普通H5）的账单明细及计费状态。

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：GetWeChatBillDetails。
Version	是	String	公共参数，本接口取值：2018-03-01。
Region	否	String	公共参数，本接口不需要传递此参数。
Date	是	Date	拉取的日期（YYYY-MM-DD）。 - 最大可追溯到365天前。 - 当天6点后才能拉取前一天的数据。 示例值：2023-03-01
Cursor	是	Integer	游标。 - 用于分页。 - 取第一页时传0，取后续页面时，传入本接口响应中返回的NextCursor字段的值。 示例值：1
RuleId	否	String	需要拉取账单详情业务对应的RuleId。 - 不传会返回所有RuleId数据。 - 默认为空字符串。 示例值：12

3. 输出参数

参数名称	类型	描述
HasNextPage	Boolean	是否还有下一页。 - 该字段为true时，需要将NextCursor的值作为入参Cursor继续调用本接口。 示例值：false
NextCursor	Integer	下一页的游标，用于分页。 示例值：1
WeChatBillDetails	Array of WeChatBillDetail	数据。
RequestId	String	唯一请求 ID，由服务端生成，每次请求都会返回（若请求因其他原因未能抵达服务端，则该次请求不会获得RequestId）。定位问题时需要提供该次请求的 RequestId。

4. 示例

示例1 获取第一页数据

获取第一页数据时，Cursor值传0。当HasNextPage的值为true时，继续调用接口。

输入示例

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

{
"Date": "2020-09-22",
"Cursor": 0,
"RuleId": "2"
}
```

输出示例

```
{
"Response": {
"NextCursor": 1873,
"WeChatBillDetails": [
{
"ChargeCount": 1,
"ChargeDetails": [
{
"ErrorMessage": "成功",
"Name": "韦小宝",
"Seq": "e594dc57-9ee3-409b-a8e4-7b46945887c3",
"IdCard": "1120416541220243X",
"ErrorCode": "0",
"IsNeedCharge": true,
"ChargeType": "核身",
"ReqTime": "1638781186273",
"Sim": "100.00"
}
],
"BizToken": "3DCE6611F1A-0F1E-455BD-BE13-34C05CEA7681",
"RuleId": "96"
}
],
"HasNextPage": true,
"RequestId": "fdce508f-0813-4229-8a2e-439b851ec99f"
}
}
```

示例2 获取后续页面数据

获取后续页面数据时，将上一页返回的NextCursor值作为入参传入。当HasNextPage的值为false的时候，即可停止调用接口。

输入示例

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

```
{
"Date": "2020-09-22",
"Cursor": 1873,
"RuleId": "2"
}
```

输出示例

```
{
"Response": {
"NextCursor": 28888,
"WeChatBillDetails": [
{
"ChargeCount": 1,
"ChargeDetails": [
{
"ErrorMessage": "成功",
"Name": "韦小宝",
"Seq": "e594dc357-9ee3-409b-a8e4-7b469458827c3",
"IdCard": "11204416541220243X",
"ErrorCode": "0",
"IsNeedCharge": true,
"ChargeType": "核身",
"ReqTime": "1638781186273",
"Sim": "100.00"
}
],
"BizToken": "FFE6212B2-8DC2-4D03-BC1D-29C5054A221F9",
"RuleId": "96"
}
],
"HasNextPage": false,
"RequestId": "a87bb0ee-e95b-415be-b562-117944c4ceb1"
}
}
```

示例3 获取后续页面数据异常示例

获取后续页面数据，输入不存在的RuleId。

输入示例

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

{
"Date": "2020-09-22",
"Cursor": 1873,
"RuleId": "23"
}
```

输出示例

```
{
"Response": {
"Error": {
"Code": "InvalidParameterValue.RuleIdNotExist",
"Message": "RuleId不存在，请到人脸核身控制台申请。"
},
"RequestId": "c5f6c7e4-2b4e-4a77-acc5-d111fffe8302"
}
}
```

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.UnKnown	内部未知错误。
InvalidParameter	参数错误。
InvalidParameterValue.RuleIdNotExist	RuleId不存在，请到人脸核身控制台申请。

获取证件NFC结果

最近更新时间：2025-05-26 19:47:54

1. 接口描述

接口请求域名： faceid.tencentcloudapi.com 。

解析SDK获取到的证件NFC数据，接口传入SDK返回的ReqId，返回证件信息（个别字段为特定证件类型特有）。SDK生成的ReqId五分钟内有效，重复查询仅收一次费。支持身份证类证件（二代身份证、港澳居住证、台湾居住证、外国人永居证）以及旅行类证件（港澳通行证、台湾通行证、台胞证、回乡证）的NFC识别及核验。

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：ParseNfcData。
Version	是	String	公共参数，本接口取值：2018-03-01。
Region	否	String	公共参数，本接口不需要传递此参数。
ReqId	是	String	前端SDK返回。 示例值：ECBEC11E6-F05A-40DE-9CC3-C9EF119F55FE1

3. 输出参数

参数名称	类型	描述
ResultCode	String	结果码。 - 取值范围：0为首次查询成功，-1为查询失败。 注意：此字段可能返回 null，表示取不到有效值。 示例值：0
IdNum	String	身份证号。 注意：此字段可能返回 null，表示取不到有效值。 示例值：112044165412202 43X
Name	String	姓名。 注意：此字段可能返回 null，表示取不到有效值。 示例值：韦小宝
Picture	String	照片。 注意：此字段可能返回 null，表示取不到有效值。 示例值：/9j/4AAQSkZJRg.....s97n//2Q==
BirthDate	String	出生日期。 注意：此字段可能返回 null，表示取不到有效值。 示例值：20220222
BeginTime	String	有效期起始时间。 注意：此字段可能返回 null，表示取不到有效值。 示例值：20220222
EndTime	String	有效期结束时间。 注意：此字段可能返回 null，表示取不到有效值。 示例值：20220222
Address	String	住址。 注意：此字段可能返回 null，表示取不到有效值。

参数名称	类型	描述
		示例值：北京市东城区景山前街4号紫禁城敬事房
Nation	String	民族。 注意：此字段可能返回 null，表示取不到有效值。 示例值：汉
Sex	String	性别。 注意：此字段可能返回 null，表示取不到有效值。 示例值：男
IdType	String	类型。 - 取值范围： 01：身份证。 03：中国护照。 04：军官证。 05：武警证。 06：港澳通行证 。 07：台胞证 。 08：外国护照 。 09：士兵证。 10：临时身份证。 11：户口本 。 12：警官证 。 13：外国人永久居留证。 14：港澳台居民居住证。 15：回乡证。 16：大陆居民来往台湾通行证。 99：其他证件。 注意：此字段可能返回 null，表示取不到有效值。 示例值：01
EnName	String	英文姓名。 注意：此字段可能返回 null，表示取不到有效值。 示例值：lily
SigningOrganization	String	签发机关 注意：此字段可能返回 null，表示取不到有效值。 示例值：广东省
OtherIdNum	String	港澳台居民居住证，通行证号码。 注意：此字段可能返回 null，表示取不到有效值。 示例值：830000199706020042
Nationality	String	旅行证件国籍。 注意：此字段可能返回 null，表示取不到有效值。 示例值：英国
PersonalNumber	String	旅行证件机读区第二行 29~42 位。 注意：此字段可能返回 null，表示取不到有效值。 示例值：0602004212071
CheckMRTD	String	旅行证件类的核验结果。 - JSON格式如下： {"result_issuer ":"签发者证书合法性验证结果 ","result_paper":"证件安全对象合法性验证结果 ","result_data" :"防数据篡改验证结果 ","result_chip" :"防证书件芯片被复制验证结果"} 。 - 取值范围： 0:验证通过，1: 验证不通过，2: 未验证，3:部分通过，当4项核验结果都为0时，表示证件为真。 注意：此字段可能返回 null，表示取不到有效值。
ImageA	String	身份证照片面合成图片。 注意：此字段可能返回 null，表示取不到有效值。 示例值：/9j/4AAQSkZJRg.....s97n//2Q==
ImageB	String	身份证国徽面合成图片。 注意：此字段可能返回 null，表示取不到有效值。 示例值：/9j/4AAQSk.....j+5QVYZKxU
ResultDescription	String	对result code的结果描述。 注意：此字段可能返回 null，表示取不到有效值。

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

4. 示例

示例1 解析SDK获取到的证件NFC数据

解析SDK获取到的证件NFC数据。

输入示例

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

{
"ReqId": "eyJyZXFJZCI6IjEyMzEiLCJkZXZpY2VJZCI6IjQ1NiJ9"
}
```

输出示例

```
{
"Response": {
"Address": "北京市东城区景山前街4号紫禁城敬事房",
"BeginTime": "20170405",
"BirthDate": "19890604",
"CheckMRTD": "",
"EnName": "",
"EndTime": "20260704",
"IdNum": "112044165412202 43X",
"IdType": "",
"ImageA": "/9j/4AAQSkZJRg.....s97n//2Q==",
"ImageB": "/9j/4AAQSk...mA7pvm5g==",
"Name": "韦小宝",
"Nation": "汉",
"Nationality": "",
"OtherIdNum": "",
"PersonalNumber": "",
"Picture": "Qk3OlwAAAA...7+/v7+/pAA",
"RequestId": "cdd7a769-d288-42a5-8e2e-6a5e5c9c08ae",
"ResultCode": "0",
"Sex": "男",
"SigningOrganization": "北京市东城区分局",
"ResultDescription": "首次查询成功"
}
}
```

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
InternalError.UnKnown	内部未知错误。
InvalidParameterValue	参数取值错误。
UnauthorizedOperation.Arrears	账号已欠费。
UnauthorizedOperation.Nonactivated	未开通服务。

数据结构

最近更新时间：2025-06-11 01:30:31

AttackRiskDetail

疑似攻击风险详情

被如下接口引用：DetectAIFakeFaces。

名称	类型	描述
Type	String	疑似的攻击痕迹类型 SuspectedSpoofingAttack：翻拍攻击 SuspectedSynthesisImage：疑似合成图片 SuspectedSynthesisVideo：疑似合成视频 SuspectedeAnomalyAttack：人脸特征疑似非真人 SuspectedAdversarialAttack：疑似对抗样本攻击 SuspectedBlackIndustry：疑似黑产批量模版攻击 SuspectedWatermark：疑似存在水印 注意：此字段可能返回 null，表示取不到有效值。 示例值：SuspectedWatermark

ChargeDetail

计费详情

被如下接口引用：GetWeChatBillDetails。

名称	类型	描述
ReqTime	String	一比一时间时间戳，13位。
Seq	String	一比一请求的唯一标记。
IdCard	String	一比一时使用的、脱敏后的身份证号。
Name	String	一比一时使用的、脱敏后的姓名。
Sim	String	一比一的相似度。0-100，保留2位小数。
IsNeedCharge	Boolean	本次详情是否收费。
ChargeType	String	收费类型，比对、核身、混合部署。
ErrorCode	String	本次活体一比一最终结果。
ErrorMessage	String	本次活体一比一最终结果描述。

DetectDetail

活体一比一详情

被如下接口引用：GetDetectInfoEnhanced, GetEidResult。

名称	类型	描述
ReqTime	String	请求时间戳。 注意：此字段可能返回 null，表示取不到有效值。
Seq	String	本次活体一比一请求的唯一标记。 注意：此字段可能返回 null，表示取不到有效值。
Idcard	String	参与本次活体一比一的身份证号。 注意：此字段可能返回 null，表示取不到有效值。
Name	String	参与本次活体一比一的姓名。 注意：此字段可能返回 null，表示取不到有效值。

名称	类型	描述
Sim	String	本次活体一比一的相似度。 注意：此字段可能返回 null，表示取不到有效值。
IsNeedCharge	Boolean	本次活体一比一是否收费 注意：此字段可能返回 null，表示取不到有效值。 示例值：false
Errcode	Integer	本次活体一比一最终结果。0为成功 注意：此字段可能返回 null，表示取不到有效值。 示例值：0
Errmsg	String	本次活体一比一最终结果描述。（仅描述用，文案更新时不会通知。） 注意：此字段可能返回 null，表示取不到有效值。
Livestatus	Integer	本次活体结果。0为成功 注意：此字段可能返回 null，表示取不到有效值。 示例值：0
Livemsg	String	本次活体结果描述。（仅描述用，文案更新时不会通知。） 注意：此字段可能返回 null，表示取不到有效值。
Comparestatus	Integer	本次一比一结果。0为成功 注意：此字段可能返回 null，表示取不到有效值。 示例值：0
Comparemsg	String	本次一比一结果描述。（仅描述用，文案更新时不会通知。） 注意：此字段可能返回 null，表示取不到有效值。
CompareLibType	String	比对库源类型。包括： 公安商业库； 业务方自有库（用户上传照片、客户的混合库、混合部署库）； 二次验证库； 人工审核库； 注意：此字段可能返回 null，表示取不到有效值。
LivenessMode	Integer	枚举活体检测类型： 0：未知 1：数字活体 2：动作活体 3：静默活体 4：一闪活体（动作+光线） 注意：此字段可能返回 null，表示取不到有效值。 示例值：0

DetectInfoBestFrame

核身最佳帧信息。

被如下接口引用：GetDetectInfoEnhanced, GetEidResult。

名称	类型	描述
BestFrame	String	活体比对最佳帧Base64编码。 注意：此字段可能返回 null，表示取不到有效值。 示例值：/9j/4AAQSk...NFFFJjP//Z
BestFrames	Array of String	自截帧Base64编码数组。 注意：此字段可能返回 null，表示取不到有效值。 示例值：["/9j/4AAQSk...7+lc5J/9k="]

DetectInfoIdCardData

核身身份证图片信息。

被如下接口引用：GetDetectInfoEnhanced, GetEidResult。

名称	类型	描述
OcrFront	String	OCR正面照片的base64编码。 注意：此字段可能返回 null，表示取不到有效值。 示例值：/9j/4AAQSkZJRg.....s97n//2Q==
OcrBack	String	OCR反面照片的base64编码。 注意：此字段可能返回 null，表示取不到有效值。 示例值：/9j/4AAQSk.../S7s2f/9k=
ProcessedFrontImage	String	旋转裁边后的正面照片base64编码。 注意：此字段可能返回 null，表示取不到有效值。 示例值：/9j/4AAQSk...PnUZx//9k=
ProcessedBackImage	String	旋转裁边后的背面照片base64编码。 注意：此字段可能返回 null，表示取不到有效值。 示例值：/9j/4AAQSk...Vr0Ej/2Q==
Avatar	String	身份证正面人像图base64编码。 注意：此字段可能返回 null，表示取不到有效值。 示例值：/9j/4AAQSk...iiiisyz//Z
WarnInfos	Array of Integer	身份证人像面告警码。 - 开启身份证告警功能后才会返回。 - 返回数组中可能出现的告警码如下： - -9100 身份证有效日期不合法告警。 - -9101 身份证边框不完整告警。 - -9102 身份证复印件告警。 - -9103 身份证翻拍告警。 - -9105 身份证框内遮挡告警。 - -9104 临时身份证告警。 - -9106 身份证 PS 告警（疑似存在PS痕迹）。 - -9107 身份证反光告警。 注意：此字段可能返回 null，表示取不到有效值。 示例值：["-9100"]
BackWarnInfos	Array of Integer	身份证国徽面告警码。 - 开启身份证告警功能后才会返回。 - 返回数组中可能出现的告警码如下： - -9100 身份证有效日期不合法告警， - -9101 身份证边框不完整告警， - -9102 身份证复印件告警， - -9103 身份证翻拍告警， - -9105 身份证框内遮挡告警， - -9104 临时身份证告警， - -9106 身份证 PS 告警（疑似存在PS痕迹）， - -9107 身份证反光告警。 注意：此字段可能返回 null，表示取不到有效值。 示例值：["-9100"]

DetectInfoText

核身文本信息。

被如下接口引用：GetDetectInfoEnhanced, GetEidResult。

名称	类型	描述
ErrCode	Integer	本次流程最终验证结果。 - 取值范围：0为成功。 - 仅包含活体人脸核身结果，不包含意愿核身结果。 注意：此字段可能返回 null，表示取不到有效值。 示例值：0
ErrMsg	String	本次流程最终验证结果描述。 - 仅描述用，文案更新时不会通知。 注意：此字段可能返回 null，表示取不到有效值。 示例值：成功

名称	类型	描述
IdCard	String	本次验证使用的身份证号。 注意：此字段可能返回 null，表示取不到有效值。 示例值：11204416541220243X
UseIDType	Integer	用户认证时使用的证件号码类型。 - 取值范围： 0：二代身份证的证件号码。 1：港澳台居住证的证件号码。 2：其他（核验使用的证件号码非合法身份号码）。 注意：此字段可能返回 null，表示取不到有效值。 示例值：0
Name	String	本次验证使用的姓名。 注意：此字段可能返回 null，表示取不到有效值。 示例值：韦小宝
OcrNation	String	身份校验环节识别结果：民族。 注意：此字段可能返回 null，表示取不到有效值。 示例值：汉族
OcrAddress	String	身份校验环节识别结果：家庭住址。 注意：此字段可能返回 null，表示取不到有效值。 示例值：北京市东城区景山前街4号紫禁城敬事房
OcrBirth	String	身份校验环节识别结果：生日。 - 格式为：YYYY/M/D 注意：此字段可能返回 null，表示取不到有效值。 示例值：1995/5/13
OcrAuthority	String	身份校验环节识别结果：签发机关。 注意：此字段可能返回 null，表示取不到有效值。 示例值：北京市
OcrValidDate	String	身份校验环节识别结果：有效日期。 - 格式为：YYYY.MM.DD-YYYY.MM.DD。 注意：此字段可能返回 null，表示取不到有效值。 示例值：1999.01.01-2009.01.01
OcrName	String	身份校验环节识别结果：姓名。 注意：此字段可能返回 null，表示取不到有效值。 示例值：韦小宝
OcrIdCard	String	身份校验环节识别结果：身份证号。 注意：此字段可能返回 null，表示取不到有效值。 示例值：11204416541220243X
OcrGender	String	身份校验环节识别结果：性别。 注意：此字段可能返回 null，表示取不到有效值。 示例值：男
IdInfoFrom	String	身份校验环节采用的信息上传方式。 - 取值有"NFC"、"OCR"、"手动输入"、"其他" 注意：此字段可能返回 null，表示取不到有效值。 示例值：NFC
LiveStatus	Integer	本次流程最终活体结果。 - 0为成功 注意：此字段可能返回 null，表示取不到有效值。 示例值：0
LiveMsg	String	本次流程最终活体结果描述。 - 仅描述用，文案更新时不会通知。 注意：此字段可能返回 null，表示取不到有效值。 示例值：成功
Comparestatus	Integer	本次流程最终一比一结果。 - 0为成功

名称	类型	描述
		注意：此字段可能返回 null，表示取不到有效值。 示例值：0
Comparemsg	String	本次流程最终一比一结果描述。 - 仅描述用，文案更新时不会通知。 注意：此字段可能返回 null，表示取不到有效值。 示例值：成功
Sim	String	本次流程活体一比一的分数。 - 取值范围 [0.00, 100.00]。 - 相似度大于等于70时才判断为同一人，也可根据具体场景自行调整阈值。 - 阈值70的误通过率为千分之一，阈值80的误通过率是万分之一。 注意：此字段可能返回 null，表示取不到有效值。 示例值：88.3
Location	String	地理位置经纬度。 注意：此字段可能返回 null，表示取不到有效值。 示例值：2
Extra	String	Auth接口带入额外信息。 注意：此字段可能返回 null，表示取不到有效值。 示例值：2
LivenessDetail	Array of DetectDetail	本次流程进行的活体一比一流水。 注意：此字段可能返回 null，表示取不到有效值。
LivenessInfoTag	Array of String	描述当前请求活体阶段被拒绝的详细原因，该参数仅限PLUS版本核身服务返回。 - 详情如下： 01-用户全程闭眼 02-用户未完成指定动作 03-疑似翻拍攻击 04-疑似合成攻击 05-疑似黑产模版 06-疑似存在水印 07-反光校验未通过 08-疑似中途换人 09-人脸质量过差 10-距离校验不通过 11-疑似对抗样本攻击 12-嘴巴区域疑似存在攻击痕迹 13-眼睛区域疑似存在攻击痕迹 14-眼睛或嘴巴被遮挡 注意：此字段可能返回 null，表示取不到有效值。 示例值：[01]
Mobile	String	手机号码。 注意：此字段可能返回 null，表示取不到有效值。 示例值：16137688175
CompareLibType	String	本次流程最终比对库源类型。 - 取值范围： 权威库。 业务方自有库（用户上传照片、客户的混合库、混合部署库）。 二次验证库。 人工审核库。 注意：此字段可能返回 null，表示取不到有效值。 示例值：权威库
LivenessMode	Integer	本次流程最终活体类型。 - 取值范围： 0：未知 1：数字活体 2：动作活体 3：静默活体 4：一闪活体（动作+光线） 5：远近活体 注意：此字段可能返回 null，表示取不到有效值。 示例值：0

名称	类型	描述
NFCRequestIds	Array of String	nfc重复计费requestId列表。 注意：此字段可能返回 null，表示取不到有效值。 示例值：["5bd763c5-3dd9-4d40-8043-00f90fe75acb"]
NFCBillingCounts	Integer	nfc重复计费计数。 注意：此字段可能返回 null，表示取不到有效值。 示例值：1
PassNo	String	港澳台居住证通行证号码。 注意：此字段可能返回 null，表示取不到有效值。 示例值：08320271
VisaNum	String	港澳台居住证签发次数。 注意：此字段可能返回 null，表示取不到有效值。 示例值：02

DetectInfoVideoData

核身视频信息。

被如下接口引用：GetDetectInfoEnhanced。

名称	类型	描述
LivenessVideo	String	活体视频的base64编码。 注意：此字段可能返回 null，表示取不到有效值。 示例值：/9j/4AAQSkZJRg.....s97n//2Q==
LivenessVideos	Array of VideoDetailData	当次token中所有用户活体视频的COS存储路径，仅当您开启数据存储服务且“IsReturnAllVideo”入参取值为true 时返回。 注意：此字段可能返回 null，表示取不到有效值。

EidInfo

Eid出参，包括商户方用户的标识和加密的用户姓名身份证信息。

被如下接口引用：GetEidResult。

名称	类型	描述
EidCode	String	商户方 appeIDcode 的数字证书。 示例值：DtROLOW1.....UJWtn+4UNOPrC10wO/bWkP0xPDAw
EidSign	String	Eid中心针对商户方EidCode的电子签名。 示例值：RFQSPB8CxI...qi5fVVyQ==
DesKey	String	商户方公钥加密的会话密钥的base64字符串，指引详见。 示例值：VDsTCWy/VU...lf26IlaWD=
UserInfo	String	会话密钥sm2加密后的base64字符串，指引详见。 示例值：Py6yFkw90h...VenILSag==

Encryption

敏感数据加密

被如下接口引用：BankCard2EVerification, BankCard4EVerification, BankCardVerification, CheckBankCardInformation, CheckIdCardInformation, CheckIdNameDate, CheckPhoneAndName, DetectAIFakeFaces, DetectAuth, GetDetectInfoEnhanced, GetEidToken, GetFaceIdResult, GetFaceIdToken, IdCardOCRVerification, IdCardVerification, ImageRecognition, ImageRecognitionV2, LivenessRecognition, MinorsVerification, MobileNetworkTimeVerification, MobileStatus, PhoneVerificationCMCC, PhoneVerificationCTCC, PhoneVerificationCUCC。

名称	类型	必选	描述
EncryptList	Array of String	是	在使用加密服务时，填入要被加密的字段。本接口中可填入加密后的一个或多个字段 注意：此字段可能返回 null，表示取不到有效值。 示例值：null
CiphertextBlob	String	是	加密后的对称密钥，关于密钥的生成和使用请查阅数据加密 文档。 注意：此字段可能返回 null，表示取不到有效值。 示例值：null
Iv	String	是	有加密需求的用户，传入CBC加密的初始向量（客户自定义字符串，长度16字符）。 注意：此字段可能返回 null，表示取不到有效值。 示例值：null
Algorithm	String	否	加密使用的算法（支持'AES-256-CBC'、'SM4-GCM'），不传默认为'AES-256-CBC' 注意：此字段可能返回 null，表示取不到有效值。 示例值：SM4-GCM
TagList	Array of String	否	SM4-GCM算法生成的消息摘要（校验消息完整性时使用） 注意：此字段可能返回 null，表示取不到有效值。 示例值：null

ExtraInfo

额外的详细信息

被如下接口引用：DetectAIFakeFaces。

名称	类型	描述
RetrievalLivenessExtraInfo	Array of RetrievalLivenessExtraInfo	命中模板的详细信息，仅返回命中的相似度最高的模板信息 注意：此字段可能返回 null，表示取不到有效值。

GetEidTokenConfig

获取token时的配置

被如下接口引用：GetEidToken。

名称	类型	必选	描述
InputType	String	否	姓名身份证输入方式。 - 取值范围： 1：传身份证正反面OCR。 2：传身份证正面OCR。 3：用户手动输入。 4：客户后台传入。 - 默认值：1。 - 注意：使用OCR时仅支持用户修改结果中的姓名。 示例值：1
UseIntentionVerify	Boolean	否	是否使用意愿核身。 - 默认不使用。 - 注意：如开启使用，则计费标签按【意愿核身】计费标签计价；如不开启，则计费标签按【E证通】计费标签计价，价格详见：价格说明。 示例值：true
IntentionMode	String	否	意愿核身模式。 - 取值范围： 1：语音朗读模式。 2：语音问答模式。 3：点头确认模式。 - 默认值为1。 示例值：1
IntentionVerifyText	String	否	意愿核身朗读模式使用的文案。 - 若未使用意愿核身朗读功能，该字段无需传入。

名称	类型	必选	描述
			- 默认为空，最长可接受120的字符串长度。 示例值："请阅读"
IntentionQuestions	Array of IntentionQuestion	否	意愿核身问答模式的配置列表。 - 问答模式支持1-10轮（不超过10轮）的意愿确认。
IntentionActions	Array of IntentionActionConfig	否	意愿核身（点头确认模式）使用的文案。 - 若未使用意愿核身（点头确认模式），则该字段无需传入。 - 默认为空，最长可接受150的字符串长度。 - 点头确认模式支持1-10轮（不超过10轮）的意愿确认。
IntentionRecognition	Boolean	否	意愿核身过程中识别用户的回答意图。 - 开启后除了IntentionQuestions的Answers列表中的标准回答会通过，近似意图的回答也会通过。 - 默认开启。 示例值：true
IsSupportHMTResidentPermitOCR	Boolean	否	是否支持港澳台居住证识别。 示例值：false
MouthOpenRecognition	Boolean	否	用户语音回答过程中是否开启张嘴识别检测。 - 默认不开启。 - 仅在意愿核身问答模式中使用。 示例值：false
Speed	Integer	否	意愿核身语音播报速度。 - 配置后问答模式和点头模式的语音播报环节都会生效。 - 默认值为0。 - 取值范围： 0：智能语速（根据播报文案的长度自动调整语音播报速度）。 1：固定1倍速。 2：固定1.2倍速。 3：固定1.5倍速。 示例值：0

IntentionActionConfig

意愿核身（点头确认模式）配置

被如下接口引用：DetectAuth, GetEidToken。

名称	类型	必选	描述
Text	String	是	点头确认模式下，系统语音播报使用的问题文本，问题最大长度为150个字符。 示例值：请问您本次业务是本人自愿办理吗？如是，请点头确认。

IntentionActionResult

意愿核身点头确认模式结果

被如下接口引用：GetDetectInfoEnhanced, GetEidResult。

名称	类型	描述
FinalResultDetailCode	Integer	意愿核身错误码： 0: "成功" -1: "参数错误" -2: "系统异常" -101: "请保持人脸在框内" -102: "检测到多张人脸" -103: "人脸检测失败" -104: "人脸检测不完整" -105: "请勿遮挡眼睛" -106: "请勿遮挡嘴巴" -107: "请勿遮挡鼻子" -201: "人脸比对相似度低" -202: "人脸比对失败"

名称	类型	描述
		-301: "意愿核验不通过" -800: "前端不兼容错误" -801: "用户未授权摄像头和麦克风权限" -802: "核验流程异常中断，请勿切屏或进行其他操作" -803: "用户主动关闭链接/异常断开链接" -804: "用户当前网络不稳定，请重试" -998: "系统数据异常" -999: "系统未知错误，请联系人工核实" 若在人脸核身过程失败、未进入意愿确认过程，则该参数返回为空，请参考人脸核身错误码结果（DetectInfoText.ErrCode） 注意：此字段可能返回 null，表示取不到有效值。 示例值：0
FinalResultMessage	String	意愿核身错误信息 注意：此字段可能返回 null，表示取不到有效值。 示例值：成功
Details	Array of IntentionActionResultDetail	意愿核身结果详细数据，与每段点头确认过程一一对应 注意：此字段可能返回 null，表示取不到有效值。

IntentionActionResultDetail

意愿核身点头确认模式结果详细数据

被如下接口引用：GetDetectInfoEnhanced, GetEidResult。

名称	类型	描述
Video	String	视频base64编码（其中包含全程提示文本和点头音频，mp4格式） 注意：此字段可能返回 null，表示取不到有效值。 示例值：AAAAIGZ0eX...0tLS0tLS8=
ScreenShot	Array of String	屏幕截图base64编码列表 注意：此字段可能返回 null，表示取不到有效值。 示例值：/9j/4AAQSkZJRg.....s97n//2Q==

IntentionQuestion

意愿核身过程中播报的问题文本、用户回答的标准文本。

被如下接口引用：DetectAuth, GetEidToken。

名称	类型	必选	描述
Question	String	是	当选择语音问答模式时，系统自动播报的问题文本。 - 最大长度为150个字符。 示例值：请问您本次业务是本人自愿办理吗？如是，请回复“我同意”。
Answers	Array of String	是	当选择语音问答模式时，用于判断用户回答是否通过的标准答案列表。 - 传入后可自动判断用户回答文本是否在标准文本列表中。 - 列表长度最大为50，单个答案长度限制10个字符。 示例值：“同意”，“我同意”，“确认”，“我确认”

IntentionQuestionResult

意愿核身问答模式结果。

被如下接口引用：GetDetectInfoEnhanced, GetEidResult。

名称	类型	描述
FinalResultDetailCode	Integer	意愿核身错误码。 - 取值范围： 0: "成功" -1: "参数错误" -2: "系统异常"

名称	类型	描述
		-101: "请保持人脸在框内" -102: "检测到多张人脸" -103: "人脸检测失败" -104: "人脸检测不完整" -105: "请勿遮挡眼睛" -106: "请勿遮挡嘴巴" -107: "请勿遮挡鼻子" -201: "人脸比对相似度低" -202: "人脸比对失败" -301: "意愿核验不通过" -302: "用户回答阶段未检测到张嘴动作" -800: "前端不兼容错误" -801: "用户未授权摄像头和麦克风权限" -802: "核验流程异常中断，请勿切屏或进行其他操作" -803: "用户主动关闭链接/异常断开链接" -804: "用户当前网络不稳定，请重试" -998: "系统数据异常" -999: "系统未知错误，请联系人工核实" - 若在人脸核身过程失败、未进入意愿确认过程，则该参数返回为空，请参考人脸核身错误码结果（DetectInfoText.ErrCode） 注意：此字段可能返回 null，表示取不到有效值。 示例值："-202"
FinalResultMessage	String	意愿核身错误信息。 注意：此字段可能返回 null，表示取不到有效值。 示例值：人脸比对失败
Video	String	视频base64。 - 其中包含全程问题和回答音频，mp4格式。 注意：此字段可能返回 null，表示取不到有效值。 示例值：/9j/4AAQSk...1fp8+5/9k=
ScreenShot	Array of String	屏幕截图base64列表。 注意：此字段可能返回 null，表示取不到有效值。 示例值：["/9j/4AAQSkZJRg.....s97n//2Q=="]
ResultCode	Array of String	和答案匹配结果列表。 - 取值范围（0：成功；-1：不匹配） 注意：此字段可能返回 null，表示取不到有效值。 示例值：0
AsrResult	Array of String	回答问题语音识别结果列表。 注意：此字段可能返回 null，表示取不到有效值。 示例值：""
Audios	Array of String	答案录音音频。 注意：此字段可能返回 null，表示取不到有效值。 示例值：""
FinalResultCode	String	意愿核身最终结果。 - 取值范围： 0：认证通过。 -1：认证未通过。 -2：浏览器内核不兼容，无法进行意愿校验。 - 建议使用“FinalResultDetailCode”参数获取详细的错误码信息。 注意：此字段可能返回 null，表示取不到有效值。 示例值：0

IntentionVerifyData

意愿核身相关结果。

被如下接口引用：GetDetectInfoEnhanced, GetEidResult。

名称	类型	描述
IntentionVerifyVideo	String	意愿确认环节中录制的视频（base64）。 - 若不存在则为空字符串。 注意：此字段可能返回 null，表示取不到有效值。 示例值：/9j/4AAQSkZJRg.....s97n//2Q==
AsrResult	String	意愿确认环节中用户语音转文字的识别结果。 - 若不存在则为空字符串。 注意：此字段可能返回 null，表示取不到有效值。 示例值：""
ErrorCode	Integer	意愿确认环节的结果码。 - 当该结果码为0时，语音朗读的视频与语音识别结果才会返回。 注意：此字段可能返回 null，表示取不到有效值。 示例值：0
ErrorMessage	String	意愿确认环节的结果信息。 注意：此字段可能返回 null，表示取不到有效值。 示例值：""
IntentionVerifyBestFrame	String	意愿确认环节中录制视频的最佳帧（base64）。 - 若不存在则为空字符串。 注意：此字段可能返回 null，表示取不到有效值。 示例值：/9j/4AAQSk...+0s8R/ef/Z

RetrievalLivenessExtraInfo

模版检索详细信息

被如下接口引用：DetectAIFakeFaces。

名称	类型	描述
HitGroup	String	命中的模版类型，其中Common-公共库；Auto-自动聚类库；Owner-自建模版库 注意：此字段可能返回 null，表示取不到有效值。 示例值：Common
SimilarityScore	Float	命中的相似度 注意：此字段可能返回 null，表示取不到有效值。 示例值：0.98
HitTemplate	String	命中的模板id 注意：此字段可能返回 null，表示取不到有效值。 示例值：45

RuleIdConfig

RuleId相关配置

被如下接口引用：DetectAuth。

名称	类型	必选	描述
IntentionRecognition	Boolean	否	意愿核身过程中识别用户的回答意图，开启后除了IntentionQuestions的Answers列表中的标准回答会通过，近似意图的回答也会通过，默认开启。 示例值：true
IntentionType	Integer	否	意愿核身类型，默认为0： 0：问答模式，DetectAuth接口需要传入IntentionQuestions字段； 1：点头模式，DetectAuth接口需要传入IntentionActions字段； 示例值：0
MouthOpenRecognition	Boolean	否	用户语音回答过程中是否开启张嘴识别检测，默认不开启，仅在意愿核身问答模式中使用。 示例值：false
Speed	Integer	否	意愿核身语音播报速度，配置后问答模式和点头模式的语音播报环节都会生效，默认值为0： 0：智能语速（根据播报文案的长度自动调整语音播报速度） 1：固定1倍速

名称	类型	必选	描述
			2：固定1.2倍速 3：固定1.5倍速 示例值：0

VideoDetailData

核身过程视频信息。

被如下接口引用：GetDetectInfoEnhanced。

名称	类型	描述
Seq	String	本次活体一比一请求的唯一标记。 注意：此字段可能返回 null，表示取不到有效值。 示例值：AE13A3B3-F276-4C65-9BC9-1FE137851581
Video	String	活体视频的base64编码。 注意：此字段可能返回 null，表示取不到有效值。 示例值：/9j/4AAQSkZJRg.....s97n//2Q==

WeChatBillDetail

账单详情

被如下接口引用：GetWeChatBillDetails。

名称	类型	描述
BizToken	String	token
ChargeCount	Integer	本token收费次数 示例值：0
ChargeDetails	Array of ChargeDetail	本token计费详情
RuleId	String	业务RuleId

错误码

最近更新时间：2024-12-24 01:35:13

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。
InvalidRequest	请求 body 的 multipart 格式错误。
IpInBlacklist	IP 地址在黑名单中。
IpNotInWhitelist	IP 地址不在白名单中。
LimitExceeded	超过配额限制。
MissingParameter	缺少参数。

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

业务错误码

错误码	说明
FailedOperation.ActionCloseEye	未检测到闭眼动作。
FailedOperation.ActionFaceClose	脸离屏幕太近。
FailedOperation.ActionFaceFar	脸离屏幕太远。
FailedOperation.ActionFaceLeft	脸离屏幕太左。
FailedOperation.ActionFaceRight	脸离屏幕太右。
FailedOperation.ActionFirstAction	未检测到动作配合。
FailedOperation.ActionLightDark	光线太暗。
FailedOperation.ActionLightStrong	光线太强。
FailedOperation.ActionNodetectFace	未能检测到完整人脸。
FailedOperation.ActionOpenMouth	未检测到张嘴动作。
FailedOperation.CompareFail	比对失败。
FailedOperation.CompareLibServiceUnavailable	比对库源维护中，暂时不可用
FailedOperation.CompareLowSimilarity	比对相似度未达到通过标准。
FailedOperation.CompareSystemError	调用比对引擎接口出错。
FailedOperation.CompressVideoError	The video compression failed. Please try again or reduce the size of the input video.
FailedOperation.CoveredFace	图中人脸存在遮挡，请传入无遮挡人脸图片

错误码	说明
FailedOperation.DbError	数据库异常。
FailedOperation.DecryptSystemError	解密失败。
FailedOperation.DetectEngineSystemError	服务引擎调用失败，请重试
FailedOperation.DownLoadError	文件下载失败。
FailedOperation.DownLoadTimeoutError	文件下载超时。
FailedOperation.EmptyImageError	图片内容为空。
FailedOperation.EncryptSystemError	加密失败。
FailedOperation.FileSaveError	文件存储失败，请稍后重试。
FailedOperation.IdFormatError	输入的身份证号有误。
FailedOperation.IdNameMisMatch	姓名和身份证号不一致，请核实后重试。
FailedOperation.IdNoExistSystem	库中无此号，请到户籍所在地进行核实。
FailedOperation.IdPhotoNoExist	库中无此号照片，请到户籍所在地进行核实。
FailedOperation.IdPhotoPoorQuality	证件图片质量差，请更新后重试。
FailedOperation.IdPhotoSystemNoanswer	客户库自建库或认证中心返照失败，请稍后再试。
FailedOperation.IdentityAuthLimitExceeded	姓名/身份证号认证次数超过当日限制，请次日重试
FailedOperation.ImageBlur	图片模糊。
FailedOperation.ImageDecodeFailed	图片解码失败。
FailedOperation.ImageNoIdCard	图片中未检测到身份证。
FailedOperation.ImageSizeTooLarge	图片尺寸过大。
FailedOperation.IncompleteFace	未检测到完整人脸，请传入完整人脸图片
FailedOperation.LifePhotoDetectFaces	检测到多张人脸。
FailedOperation.LifePhotoDetectFake	实人比对没通过。
FailedOperation.LifePhotoDetectNoFaces	未能检测到完整人脸。
FailedOperation.LifePhotoPoorQuality	传入图片分辨率太低，请重新上传。
FailedOperation.LifePhotoSizeError	传入图片过大或过小。
FailedOperation.LipFaceIncomplete	脸部未完整露出。
FailedOperation.LipMoveSmall	嘴唇动作幅度过小。
FailedOperation.LipNetFailed	视频拉取失败，请重试。
FailedOperation.LipSizeError	视频为空，或大小不合适，请控制录制时长在6s左右。
FailedOperation.LipVideoInvalid	视频格式有误。
FailedOperation.LipVideoQuaility	视频像素太低。
FailedOperation.LipVoiceDetect	未检测到声音。
FailedOperation.LipVoiceLow	视频声音太小。
FailedOperation.LipVoiceRecognize	声音识别失败。
FailedOperation.LivessBestFrameError	人脸检测失败，无法提取比对照。
FailedOperation.LivessDetectFail	活体检测没通过。
FailedOperation.LivessDetectFake	疑似非真人录制。

错误码	说明
FailedOperation.LivessSystemError	调用活体引擎接口出错。
FailedOperation.LivessUnknownError	视频实人检测没通过。
FailedOperation.NameFormatError	输入的姓名有误。
FailedOperation.OcrFailed	Ocr识别失败。
FailedOperation.PoorImageQuality	图片质量过差，请检查图片质量
FailedOperation.RequestLimitExceeded	调用次数超出限制。
FailedOperation.SilentDetectFail	实人检测失败。
FailedOperation.SilentEyeLiveFail	眼睛检测失败。
FailedOperation.SilentFaceDetectFail	视频未检测到人脸。
FailedOperation.SilentFaceQualityFail	视频中人脸质量低。
FailedOperation.SilentFaceWithMaskFail	检测到带口罩。
FailedOperation.SilentMouthLiveFail	嘴巴检测失败。
FailedOperation.SilentMultiFaceFail	视频检测中有多个人脸。
FailedOperation.SilentPictureLiveFail	疑似翻拍。
FailedOperation.SilentThreshold	实人检测未达到通过标准。
FailedOperation.SilentTooShort	视频录制时间过短，请录制2秒以上的视频。
FailedOperation.StsUnAuthErrError	STS未授权。
FailedOperation.UnKnown	内部未知错误。
FailedOperation.UnregisteredEid	该用户未注册E证通，请先注册并跟权威库核验。
FailedOperation.VerificationFail	认证不通过。
FailedOperation.VideoDecodeFailed	视频解码异常
FailedOperation.VideoDurationExceeded	视频时长过长，当前接口最大支持的视频时长为20s。
InternalError.EncryptSystemError	加密失败。
InternalError.UnKnown	内部未知错误。
InvalidParameter.RuleId	RuleId不存在。
InvalidParameter.UnsupportEncryptField	存在不加密的字段，请参考文档修改。
InvalidParameterValue.BizTokenExpired	BizToken过期。
InvalidParameterValue.BizTokenIllegal	BizToken不合法。
InvalidParameterValue.RuleIdDisabled	该ruleid已被您停用，请确认后重试。
InvalidParameterValue.RuleIdNotExist	RuleId不存在，请到人脸核身控制台申请。
UnauthorizedOperation.ActivateError	服务开通异常。
UnauthorizedOperation.Activating	服务开通中。
UnauthorizedOperation.Arrears	账号已欠费。
UnauthorizedOperation.ChargeStatusException	计费状态异常。
UnauthorizedOperation.NonAuthorize	账号未实名。
UnauthorizedOperation.Nonactivated	未开通服务。