# Face Fusion

# API Documentation

# Product Documentation

# Contents

# API Documentation

# API Category

Last updated：2024-11-08 15:17:55

## Image Face Fusion (Basic) APIs

| API Name | Feature | Frequency Limit (maximum requests per second) |
| --- | --- | --- |
| FuseFace | Image Face Fusion (Basic) | - |

## Video Face Fusion (Single Face) APIs

| API Name | Feature | Frequency Limit (maximum requests per second) |
| --- | --- | --- |
| SubmitVideoFaceFusionJob | Submit the face fusion task | - |
| QueryVideoFaceFusionJob | Queries video face fusion tasks | - |

# Making API Requests
# Request Structure

Last updated：2024-11-26 11:44:22

## 1. Service Address

The API supports access from either a nearby region (at facefusion.intl.tencentcloudapi.com) or a specified region (at facefusion.ap-guangzhou.tencentcloudapi.com for Guangzhou, for example).

We recommend using the domain name to access the nearest server. When you call an API, the request is automatically resolved to a server in the region **nearest** to the location where the API is initiated. For example, when you initiate an API request in Guangzhou, this domain name is automatically resolved to a Guangzhou server, the result is the same as that of specifying the region in the domain like "facefusion.ap-guangzhou.tencentcloudapi.com".

**Note: For latency-sensitive businesses, we recommend that you specify the region in the domain name.**

Tencent Cloud currently supports the following regions:

| Hosted region | Domain name |
| --- | --- |
| Local access region (recommended, only for non-financial availability zones) | facefusion.intl.tencentcloudapi.com |
| South China (Guangzhou) | facefusion.ap-guangzhou.tencentcloudapi.com |
| East China (Shanghai) | facefusion.ap-shanghai.tencentcloudapi.com |
| North China (Beijing) | facefusion.ap-beijing.tencentcloudapi.com |
| Southwest China (Chengdu) | facefusion.ap-chengdu.tencentcloudapi.com |
| Southwest China (Chongqing) | facefusion.ap-chongqing.tencentcloudapi.com |
| Hong Kong, Macao, Taiwan (Hong Kong, China) | facefusion.ap-hongkong.tencentcloudapi.com |
| Southeast Asia (Singapore) | facefusion.ap-singapore.tencentcloudapi.com |

| | |
|---|---|
| Southeast Asia (Bangkok) | facefusion.ap-bangkok.tencentcloudapi.com |
| South Asia (Mumbai) | facefusion.ap-mumbai.tencentcloudapi.com |
| Northeast Asia (Seoul) | facefusion.ap-seoul.tencentcloudapi.com |
| Northeast Asia (Tokyo) | facefusion.ap-tokyo.tencentcloudapi.com |
| U.S. East Coast (Virginia) | facefusion.na-ashburn.tencentcloudapi.com |
| U.S. West Coast (Silicon Valley) | facefusion.na-siliconvalley.tencentcloudapi.com |
| Europe (Frankfurt) | facefusion.eu-frankfurt.tencentcloudapi.com |

## 2. Communications Protocol

All the Tencent Cloud APIs communicate via HTTPS, providing highly secure communication tunnels.

## 3. Request Methods

Supported HTTP request methods:

- POST (recommended)
- GET

The Content-Type types supported by POST requests:

- application/json (recommended). The TC3-HMAC-SHA256 signature algorithm must be used.
- application/x-www-form-urlencoded. The HmacSHA1 or HmacSHA256 signature algorithm must be used.
- multipart/form-data (only supported by certain APIs). You must use TC3-HMAC-SHA256 to calculate the signature.

The size of a GET request packet is up to 32 KB. The size of a POST request is up to 1 MB when the HmacSHA1 or HmacSHA256 signature algorithm is used, and up to 10 MB when TC3-HMAC-SHA256 is used.

## 4. Character Encoding

Only UTF-8 encoding is used.

# Common Params

Last updated：2024-11-26 11:44:23

Common parameters are used for all APIs authenticating requestors. Common parameters must be included in all API requests, and they will not be described in individual API documents.

The exact contents of the common parameters will vary depending on the version of the signature method you use.

## Common parameters for Signature Algorithm v3

When the TC3-HMAC-SHA256 algorithm is used, the common parameters should be uniformly placed in the HTTP request header, as shown below:

| Parameter Name | Type | Required | Description |
| --- | --- | --- | --- |
| X-TC-Action | String | Yes | The name of the API for the desired operation. For the specific value, see description of common parameter `Action` in the input parameters in r documentation. For example, the API for querying the CVM instance list i `DescribeInstances` . |
| X-TC-Region | String | Yes | Region parameter, which is used to identify the region to which the data y work with belongs. For values supported for an API, see the description c parameter `Region` in the input parameters in related API documentati parameter is not required for some APIs (which will be indicated in relate documentation), and will not take effect even it is passed. |
| X-TC-Timestamp | Integer | Yes | The current UNIX timestamp that records the time when the API request for example, 1529223702. Note: If the difference between the UNIX times server time is greater than 5 minutes, a signature expiration error may oc |
| X-TC-Version | String | Yes | API version of the action. For the valid values, see the description of the c parameter `Version` in the API documentation. For example, the versi 2017-03-12. |
| Authorization | String | Yes | The HTTP authentication request header, for example: TC3-HMAC-SHA256 Credential=AKID*******/Date/service/tc3_request, SignedHeaders=content-type;host, Signature=fe5f80f77d5fa3beca038a248ff027d0445342fe2855ddc96317 Here: - TC3-HMAC-SHA256: Signature method, currently fixed as this value; - Credential: Signature credential; AKID******* is the SecretId; Date is a d time, and this value must match the value of X-TC-Timestamp (a commo |

| | | | |
|---|---|---|---|
| | | | UTC time format; service is the name of the product/service, and is gener name prefix. For example, a domain name cvm.tencentcloudapi.com refe product and the value would be cvm;<br>- SignedHeaders: The headers that contains the authentication informatic type and host are the required headers;<br>- Signature: Signature digest. |
| X-TC-Token | String | No | The token used for a temporary certificate. It must be used with a tempor can obtain the temporary key and token by calling a CAM API. No token i a long-term key. |

Assuming you want to query the list of Cloud Virtual Machine instances in the Guangzhou region, the request structure in the form of request URL, request header and request body may be as follows:

Example of an HTTP GET request structure:

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

Authorization: TC3-HMAC-SHA256 Credential=AKID********************************/20
18-10-09/cvm/tc3_request, SignedHeaders=content-type;host, Signature=5da7a33f6993
f0614b047e5df4582db9e9bf4672ba50567dba16c6ccf174c474
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
```

The following example shows you how to structure an HTTP POST (application/json) request:

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

Authorization: TC3-HMAC-SHA256 Credential=AKID********************************/20
18-05-30/cvm/tc3_request, SignedHeaders=content-type;host, Signature=582c400e06b5
924a6f2b5d7d672d79c15b13162d9279b0855cfba6789a8edb4c
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}
```

Example of an HTTP POST (multipart/form-data) request structure (only supported by specific APIs):

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

Authorization: TC3-HMAC-SHA256 Credential=AKID********************************/20
18-05-30/cvm/tc3_request, SignedHeaders=content-type;host, Signature=582c400e06b5
924a6f2b5d7d672d79c15b13162d9279b0855cfba6789a8edb4c
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--
```

## Common parameters for Signature Algorithm v1

To adopt the HmacSHA1 and HmacSHA256 signature methods, common parameters must be put into the request string, as shown below:

| Parameter Name | Type | Required | Description |
| --- | --- | --- | --- |
| Action | String | Yes | The name of the API for the desired operation. For the specific value, see the description of common parameter `Action` in the input parameters in related API documentation. For example, the API for querying the CVM instance list is `DescribeInstances`. |
| Region | String | Yes | Region parameter, which is used to identify the region to which the data you want to work with belongs. For values supported for an API, see the description of common parameter `Region` in the input parameters in related API documentation. Note: This parameter is not required for some APIs (which will be indicated in related API documentation), and will not take effect even if it is passed. |
| | | | |

| | | | |
|---|---|---|---|
| Timestamp | Integer | Yes | The current UNIX timestamp that records the time when the API request was initiated, for example, 1529223702. If the difference between the value and the current system time is too large, a signature expiration error may occur. |
| Nonce | Integer | Yes | A random positive integer used along with `Timestamp` to prevent replay attacks. |
| SecretId | String | Yes | The identifying SecretId obtained on the Cloud API Key page. A SecretId corresponds to a unique SecretKey which is used to generate the request signature (Signature). |
| Signature | String | Yes | Request signature used to verify the validity of this request. This is calculated based on the actual input parameters. For more information about how this is calculated, see the API authentication documentation. |
| Version | String | Yes | API version of the action. For the valid values, see the description of the common input parameter `Version` in the API documentation. For example, the version of CVM is 2017-03-12. |
| SignatureMethod | String | No | Signature method. Currently, only HmacSHA256 and HmacSHA1 are supported. The HmacSHA256 algorithm is used to verify the signature only when this parameter is specified as HmacSHA256. In other cases, the signature is verified with HmacSHA1. |
| Token | String | No | The token used for a temporary certificate. It must be used with a temporary key. You can obtain the temporary key and token by calling a CAM API. No token is required for a long-term key. |

Assuming you want to query the list of Cloud Virtual Machine instances in the Guangzhou region, the request structure in the form of request URL, request header and request body may be as follows:

Example of an HTTP GET request structure:

```
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
Content-Type: application/x-www-form-urlencoded
```

Example of an HTTP POST request structure:

```
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=37ac2f4fde00b0ac9bd9eadeb459b1bbee224158d66e7ae5fcadb70b2d18
1d02&Region=ap-guangzhou&Nonce=23823223&SecretId=AKID**************************
****
```

## Region List

The supported Region field values for all APIs in this product are listed as below. For any API that does not support any of the following regions, this field will be described additionally in the relevant API document.

| Region | Value |
| --- | --- |
| Southeast Asia (Singapore) | ap-singapore |

# Signature v3

Last updated：2024-11-26 11:44:25

TencentCloud API authenticates every single request, i.e., the request must be signed using the security credentials in the designated steps. Each request has to contain the signature information (Signature) in the common request parameters and be sent in the specified way and format.

## Applying for Security Credentials

The security credential used in this document is a key, which includes a SecretId and a SecretKey. Each user can have up to two pairs of keys.

- SecretId: Used to identify the API caller, which is just like a username.
- SecretKey: Used to authenticate the API caller, which is just like a password.
- **You must keep your security credentials private and avoid disclosure; otherwise, your assets may be compromised. If they are disclosed, please disable them as soon as possible.**

You can apply for the security credentials through the following steps:

1. Log in to the Tencent Cloud Console.
2. Go to the TencentCloud API Key console page.
3. On the TencentCloud API Key page, click **Create** to create a SecretId/SecretKey pair.

## Using the Resources for Developers

TencentCloud API comes with SDKs for seven commonly used programming languages, including Python, Java, PHP, Go, NodeJS and .NET. In addition, it provides API Explorer which enables online call, signature verification, and SDK code generation. If you have any troubles calculating a signature, consult these resources.

## TC3-HMAC-SHA256 Signature Algorithm

Compatible with the previous HmacSHA1 and HmacSHA256 signature algorithms, the TC3-HMAC-SHA256 signature algorithm is more secure and supports larger requests and JSON format with better performance. We recommend using TC3-HMAC-SHA256 to calculate the signature.

TencentCloud API supports both GET and POST requests. For the GET method, only the Content-Type: application/x-www-form-urlencoded protocol format is supported. For the POST method, two protocol formats,

Content-Type: application/json and Content-Type: multipart/form-data, are supported. The JSON format is supported by default for all business APIs, and the multipart format is supported only for specific business APIs. In this case, the API cannot be called in JSON format. See the specific business API documentation for more information. The POST method is recommended, as there is no difference in the results of both the methods, but the GET method only supports request packets up to 32 KB.

The following uses querying the list of CVM instances in the Guangzhou region as an example to describe the steps of signature splicing. We chose this API because:

1. CVM is activated by default, and this API is often used;
2. It is read-only and does not change the status of existing resources;
3. It covers many types of parameters, which allows it to be used to demonstrate how to use arrays containing data structures.

In the example, we try to choose common parameters and API parameters that are prone to mistakes. When you actually call an API, please use parameters based on the actual conditions. The parameters vary by API. Do not copy the parameters and values in this example.

Assuming that your SecretId and SecretKey are `AKID********************************` and `********************************`, respectively, if you want to view the status of the instance in the Guangzhou region whose CVM instance name is "unnamed" and have only one data entry returned, then the request may be:

```
curl -X POST https://cvm.tencentcloudapi.com \
-H "Authorization: TC3-HMAC-SHA256 Credential=AKID********************************
*/2019-02-25/cvm/tc3_request, SignedHeaders=content-type;host, Signature=ca282b0a
56549857d53b2beb08b0c35871c892d42d09ae30b38d456e09ce291f" \
-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": ["unnamed"], "Name": "instance-name"}]}'
```

The signature calculation process is explained in detail below.

### 1. Concatenating the CanonicalRequest String

Concatenate the canonical request string (CanonicalRequest) in the following pseudocode format:

```
CanonicalRequest =
HTTPRequestMethod + '\n' +
CanonicalURI + '\n' +
```

```
CanonicalQueryString + '\n' +
CanonicalHeaders + '\n' +
SignedHeaders + '\n' +
HashedRequestPayload
```

| Field Name | Explanation |
| --- | --- |
| HTTPRequestMethod | HTTP request method (GET or POST). This example uses `POST` . |
| CanonicalURI | URI parameter. Slash ("/") is used for API 3.0. |
| CanonicalQueryString | The query string in the URL of the originating HTTP request. This is always an empty st “” for POST requests, and is the string after the question mark (?) for GET requests. Fo example: Limit=10&Offset=0。<br>Note: `CanonicalQueryString` must be URL-encoded, referencing RFC3986, the UTF8 character set. We recommend using the programming language library. All speci characters must be encoded and capitalized. |
| CanonicalHeaders | Header information for signature calculation, including at least two headers of `host` `content-type` . Custom headers can be added to participate in the signature proce improve the uniqueness and security of the request.<br>Concatenation rules:<br>1. Both the key and value of the header should be converted to lowercase with the lead and trailing spaces removed, so they are concatenated in the format of key:value\n format;<br>2. If there are multiple headers, they should be sorted in ASCII ascending order by the header keys (lowercase).<br><br>The calculation result in this example is `content-type:application/json; charset=utf-8\nhost:cvm.tencentcloudapi.com\n` .<br>Note: `content-type` must match the actually sent content. In some programming languages, a charset value would be added even if it is not specified. In this case, the request sent is different from the one signed, and the sever will return an error indicating signature verification failed. |
| SignedHeaders | Header information for signature calculation, indicating which headers of the request participate in the signature process (they must each individually correspond to the head in CanonicalHeaders). `Content-type` and `host` are required headers.<br>Concatenation rules:<br>1. Both the key and value of the header should be converted to lowercase;<br>2. If there are multiple headers, they should be sorted in ASCII ascending order by the header keys (lowercase) and separated by semicolons (;).<br><br>The value in this example is `content-type;host` |
| HashedRequestPayload | Hash value of the request payload (i.e., the body, such as `{"Limit": 1, "Filter` |

| | |
|---|---|
| | `[{"Values": ["unnamed"], "Name": "instance-name"}]` in this exampl<br>The pseudocode for calculation is Lowercase(HexEncode(Hash.SHA256(RequestPayload))) by SHA256 hashing the pay of the HTTP request, performing hexadecimal encoding, and finally converting the enco string to lowercase letters. For GET requests, `RequestPayload` is always an empt string. The calculation result in this example is `99d58dfbc6745f6747f36bfca17dee5e6881dc0428a0a36f96199342bc5b49` |

According to the rules above, the `CanonicalRequest` string obtained in the example is as follows:

```
POST
/

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

content-type;host
99d58dfbc6745f6747f36bfca17dee5e6881dc0428a0a36f96199342bc5b4907
```

### 2. Concatenating the String to Be Signed

The string to sign is concatenated as follows:

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

| Field Name | Explanation |
|---|---|
| Algorithm | Signature algorithm, which is currently always `TC3-HMAC-SHA256`. |
| RequestTimestamp | Request timestamp, i.e., the value of the common parameter `X-TC-Timestamp` in request header, which is the UNIX timestamp of the current time in seconds, such as `1551113065` in this example. |
| CredentialScope | Scope of the credential in the format of `Date/service/tc3_request`, including date, requested service and termination string (tc3_request). **`Date` is a date in UT time, whose value should match the UTC date converted by the common parameter `X-TC-Timestamp`**; `service` is the product name, which should m the domain name of the product called. The calculation result in this example is `2019-02-25/cvm/tc3_request`. |
| | |

| | |
|---|---|
| HashedCanonicalRequest | Hash value of the CanonicalRequest string concatenated in the steps above. The pseudocode for calculation is Lowercase(HexEncode(Hash.SHA256(CanonicalRequ The calculation result in this example is `2815843035062fffda5fd6f2a44ea8a34818b0dc46f024b8b3786976a3ad` |

Note:

1. Date has to be calculated from the timestamp "X-TC-Timestamp" and the time zone is UTC+0. If you add the system's local time zone information (such as UTC+8), calls can succeed both day and night but will definitely fail at 00:00. For example, if the timestamp is 1551113065 and the time in UTC+8 is 2019-02-26 00:44:25, the UTC+0 date in the calculated Date value should be 2019-02-25 instead of 2019-02-26.
2. Timestamp must be the same as your current system time, and your system time and standard time must be synced; if the difference between Timestamp and your current system time is larger than five minutes, the request will fail. If your system time is out of sync with the standard time for a while, the request will fail and return a signature expiration error.

According to the preceding rules, the string to be signed obtained in the example is as follows:

```
TC3-HMAC-SHA256
1551113065
2019-02-25/cvm/tc3_request
2815843035062fffda5fd6f2a44ea8a34818b0dc46f024b8b3786976a3adda7a
```

### 3. Calculating the Signature

1. Calculate the derived signature key with the following pseudocode:

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

| Field Name | Explanation |
|---|---|
| SecretKey | The original SecretKey, i.e., `Gu5t9xGARNpq86cd98joQYCN3*******` . |
| Date | The Date field information in `Credential` , such as `2019-02-25` in this example. |

| | |
|---|---|
| Service | Value in the Service field in `Credential`, such as `cvm` in this example. |

2. Calculate the signature with the following pseudocode:

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

### 4. Concatenating the Authorization

The Authorization is concatenated as follows:

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

| Field Name | Explanation |
|---|---|
| Algorithm | Signature algorithm, which is always `TC3-HMAC-SHA256`. |
| SecretId | The SecretId in the key pair, i.e., `AKID********************************`. |
| CredentialScope | Credential scope (see above). The calculation result in this example is `2019-02-25/cvm/tc3_request`. |
| SignedHeaders | Header information for signature calculation (see above), such as `content-type;host` in this example. |
| Signature | Signature value. The calculation result in this example is `ca282b0a56549857d53b2beb08b0c35871c892d42d09ae30b38d456e09ce291f`. |

According to the rules above, the value obtained in the example is:

```
TC3-HMAC-SHA256 Credential=AKID********************************/2019-02-25/cvm/tc
3_request, SignedHeaders=content-type;host, Signature=ca282b0a56549857d53b2beb08b
0c35871c892d42d09ae30b38d456e09ce291f
```

The following example shows a finished authorization header:

```
POST https://cvm.tencentcloudapi.com/
Authorization: TC3-HMAC-SHA256 Credential=AKID********************************/20
19-02-25/cvm/tc3_request, SignedHeaders=content-type;host, Signature=ca282b0a5654
9857d53b2beb08b0c35871c892d42d09ae30b38d456e09ce291f
```

```
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": ["unnamed"], "Name": "instance-name"}]}
```

### 5. Signature Demo

When calling API 3.0, you are recommended to use the corresponding Tencent Cloud SDK 3.0 which encapsulates the signature process, enabling you to focus on only the specific APIs provided by the product when developing. See SDK Center for more information. Currently, the following programming languages are supported:

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

To further explain the signing process, we will use a programming language to implement the process described above. The request domain name, API and parameter values in the sample are used here. This goal of this example is only to provide additional clarification for the signature process, please see the SDK for actual usage.

The final output URL might be: https://cvm.tencentcloudapi.com/?Action=DescribeInstances&InstanceIds.0=ins-09dx96dg&Limit=20&Nonce=11886&Offset=0&Region=ap-guangzhou&SecretId=AKID**********&Signature=EliP9YW3pW28FpsEdkXt%2F%2BWcGeI%3D&Timestamp=1465185768&Version=2017-03-12.

Note: The key in the example is fictitious, and the timestamp is not the current time of the system, so if this URL is opened in the browser or called using commands such as curl, an authentication error will be returned: Signature expired. In order to get a URL that can work properly, you need to replace the SecretId and SecretKey in the example with your real credentials and use the current time of the system as the Timestamp.

Note: In the example below, even if you use the same programming language, the order of the parameters in the URL may be different for each execution. However, the order does not matter, as long as all the parameters are included in the URL and the signature is calculated correctly.

Note: The following code is only applicable to API 3.0. It cannot be directly used in other signature processes. Even with an older API, signature calculation errors may occur due to the differences in details. Please refer to the corresponding documentation.

### Java

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

public class TencentCloudAPITC3Demo {
private final static Charset UTF8 = StandardCharsets.UTF_8;
private final static String SECRET_ID = "AKID********************************";
private final static String 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");
// Pay attention to the time zone; otherwise, errors may occur
sdf.setTimeZone(TimeZone.getTimeZone("UTC"));
String date = sdf.format(new Date(Long.valueOf(timestamp + "000")));

// ************ Step 1: Concatenate the CanonicalRequest string ************
```

```
String httpRequestMethod = "POST";
String canonicalUri = "/";
String canonicalQueryString = "";
String canonicalHeaders = "content-type:application/json; charset=utf-8\n" + "hos
t:" + host + "\n";
String signedHeaders = "content-type;host";

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

// ************* Step 2: Concatenate the string to sign *************
String credentialScope = date + "/" + service + "/" + "tc3_request";
String hashedCanonicalRequest = sha256Hex(canonicalRequest);
String stringToSign = algorithm + "\n" + timestamp + "\n" + credentialScope +
"\n" + hashedCanonicalRequest;
System.out.println(stringToSign);

// ************* Step 3: Calculate the signature *************
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, string
ToSign)).toLowerCase();
System.out.println(signature);

// ************* Step 4: Concatenate the Authorization *************
String authorization = algorithm + " " + "Credential=" + SECRET_ID + "/" + creden
tialScope + ", "
+ "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

# Key Parameters
secret_id = "AKID********************************"
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": [{"Name": "instance-name", "Values": ["unnamed"
]}]}

# ************* Step 1: Concatenate the CanonicalRequest string *************
http_request_method = "POST"
canonical_uri = "/"
canonical_querystring = ""
ct = "application/json; charset=utf-8"
payload = json.dumps(params)
canonical_headers = "content-type:%s\nhost:%s\n" % (ct, host)
signed_headers = "content-type;host"
hashed_request_payload = hashlib.sha256(payload.encode("utf-8")).hexdigest()
canonical_request = (http_request_method + "\n" +
canonical_uri + "\n" +
```

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

# ************* Step 2: Concatenate the string to sign *************
credential_scope = date + "/" + service + "/" + "tc3_request"
hashed_canonical_request = hashlib.sha256(canonical_request.encode("utf-8")).hexd
igest()
string_to_sign = (algorithm + "\n" +
str(timestamp) + "\n" +
credential_scope + "\n" +
hashed_canonical_request)
print(string_to_sign)


# ************* Step 3: Calculate the Signature *************
# Function for computing signature digest
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.sha2
56).hexdigest()
print(signature)

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

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

**Golang**

```
package main

import (
"crypto/hmac"
"crypto/sha256"
"encoding/hex"
"fmt"
"time"
)

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

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

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

// step 1: build canonical request string
httpRequestMethod := "POST"
canonicalURI := "/"
canonicalQueryString := ""
canonicalHeaders := "content-type:application/json; charset=utf-8\n" + "host:" +
host + "\n"
signedHeaders := "content-type;host"
payload := `{"Limit": 1, "Filters": [{"Values": ["unnamed"], "Name": "instance-na
me"}]}`
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
<?php
$secretId = "AKID********************************";
$secretKey = "********************************";
$host = "cvm.tencentcloudapi.com";
$service = "cvm";
$version = "2017-03-12";
$action = "DescribeInstances";
$region = "ap-guangzhou";
// $timestamp = time();
$timestamp = 1551113065;
$algorithm = "TC3-HMAC-SHA256";

// step 1: build canonical request string
$httpRequestMethod = "POST";
$canonicalUri = "/";
$canonicalQueryString = "";
$canonicalHeaders = "content-type:application/json; charset=utf-8\n"."host:".$hos
t."\n";
$signedHeaders = "content-type;host";
$payload = '{"Limit": 1, "Filters": [{"Values": ["unnamed"], "Name": "instance-na
me"}]}';
$hashedRequestPayload = hash("SHA256", $payload);
$canonicalRequest = $httpRequestMethod."\n"
.$canonicalUri."\n"
.$canonicalQueryString."\n"
.$canonicalHeaders."\n"
.$signedHeaders."\n"
.$hashedRequestPayload;
echo $canonicalRequest.PHP_EOL;

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

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

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

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

**Ruby**

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

# Key Parameters
secret_id = 'AKID********************************'
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')

# ************* Step 1: Concatenate the CanonicalRequest string *************
http_request_method = 'POST'
canonical_uri = '/'
canonical_querystring = ''
```

```
canonical_headers = "content-type:application/json; charset=utf-8\nhost:#{host}
\n"
signed_headers = 'content-type;host'
# params = { 'Limit' => 1, 'Filters' => [{ 'Name' => 'instance-name', 'Values' =>
['unnamed'] }] }
# 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": ["unnamed"], "Name": "instance-nam
e"}]}'
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

# ************* Step 2: Concatenate the string to sign *************
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

# ************* Step 3: Calculate the Signature *************
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

# ************* Step 4: Concatenate the Authorization *************
authorization = "#{algorithm} Credential=#{secret_id}/#{credential_scope}, Signed
Headers=#{signed_headers}, Signature=#{signature}"
puts authorization

puts 'curl -X POST ' + endpoint \
```

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

**DotNet**

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

public class Application
{
public static string SHA256Hex(string s)
{
using (SHA256 algo = SHA256.Create())
{
byte[] hashbytes = algo.ComputeHash(Encoding.UTF8.GetBytes(s));
StringBuilder builder = new StringBuilder();
for (int i = 0; i < hashbytes.Length; ++i)
{
builder.Append(hashbytes[i].ToString("x2"));
}
return builder.ToString();
}
}
public static byte[] HmacSHA256(byte[] key, byte[] msg)
{
using (HMACSHA256 mac = new HMACSHA256(key))
{
return mac.ComputeHash(msg);
}
}

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

```
dpointRounding.AwayFromZero) / 1000;
// ************* Step 1: Concatenate the CanonicalRequest string *************
string algorithm = "TC3-HMAC-SHA256";
string httpRequestMethod = "POST";
string canonicalUri = "/";
string canonicalQueryString = "";
string contentType = "application/json";
string canonicalHeaders = "content-type:" + contentType + "; charset=utf-8\n" +
"host:" + endpoint + "\n";
string signedHeaders = "content-type;host";
string hashedRequestPayload = SHA256Hex(requestPayload);
string canonicalRequest = httpRequestMethod + "\n"
+ canonicalUri + "\n"
+ canonicalQueryString + "\n"
+ canonicalHeaders + "\n"
+ signedHeaders + "\n"
+ hashedRequestPayload;
Console.WriteLine(canonicalRequest);
Console.WriteLine("-----------------------------------");

// ************* Step 2: Concatenate the string to sign *************
string credentialScope = datestr + "/" + service + "/" + "tc3_request";
string hashedCanonicalRequest = SHA256Hex(canonicalRequest);
string stringToSign = algorithm + "\n" + requestTimestamp.ToString() + "\n" + cre
dentialScope + "\n" + hashedCanonicalRequest;
Console.WriteLine(stringToSign);
Console.WriteLine("-----------------------------------");

// ************* Step 3: Calculate the signature *************
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_requ
est"));
byte[] signatureBytes = HmacSHA256(secretSigning, Encoding.UTF8.GetBytes(stringTo
Sign));
string signature = BitConverter.ToString(signatureBytes).Replace("-", "").ToLower
();
Console.WriteLine(signature);
Console.WriteLine("-----------------------------------");

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

```
Console.WriteLine("-----------------------------------");

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)
{
// SecretID and SecretKey
string SECRET_ID = "AKID********************************";
string SECRET_KEY = "********************************";

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

// The timestamp `2019-02-26 00:44:25` used here is only for reference. In a proj
ect, use the following parameter:
// DateTime date = DateTime.UtcNow;
// Enter the correct time zone. We recommend using UTC timestamp to avoid errors.
DateTime date = new DateTime(1970, 1, 1, 0, 0, 0, 0, DateTimeKind.Utc).AddSeconds
(1551113065);
string requestPayload = "{\"Limit\": 1, \"Filters\": [{\"Values\": [\"\\u672a\\u5
47d\\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(){

const SECRET_ID = "AKID********************************"
const 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
const date = getDate(timestamp)

// ************* Step 1: Concatenate the CanonicalRequest string *************
const signedHeaders = "content-type;host"

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

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

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

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

// ************* Step 2: Concatenate the string to sign *************
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)
console.log("----------------------------")

// ************* Step 3: Calculate the signature *************
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)
console.log("----------------------------")

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

const Call_Information = '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(Call_Information)
}
main()
```

**C++**

```
#include <iostream>
#include <iomanip>
#include <sstream>
#include <string>
#include <stdio.h>
#include <time.h>
#include <openssl/sha.h>
#include <openssl/hmac.h>

using namespace std;

string get_data(int64_t &timestamp)
{
string utcDate;
char buff[20] = {0};
// time_t timenow;
struct tm sttime;
sttime = *gmtime(&timestamp);
strftime(buff, sizeof(buff), "%Y-%m-%d", &sttime);
utcDate = string(buff);
return utcDate;
}
string int2str(int64_t n)
{
std::stringstream ss;
ss << n;
return ss.str();
}
string sha256Hex(const string &str)
{
char buf[3];
unsigned char hash[SHA256_DIGEST_LENGTH];
SHA256_CTX sha256;
SHA256_Init(&sha256);
SHA256_Update(&sha256, str.c_str(), str.size());
SHA256_Final(hash, &sha256);
std::string NewString = "";
for(int i = 0; i < SHA256_DIGEST_LENGTH; i++)
{
snprintf(buf, sizeof(buf), "%02x", hash[i]);
NewString = NewString + buf;
}
return NewString;
```

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

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

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

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

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

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

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

```
return output;
}

int main()
{
string SECRET_ID = "AKID********************************";
string 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);

// ************* Step 1: Concatenate the CanonicalRequest string *************
string httpRequestMethod = "POST";
string canonicalUri = "/";
string canonicalQueryString = "";
string canonicalHeaders = "content-type:application/json; charset=utf-8\nhost:" +
host + "\n";
string signedHeaders = "content-type;host";
string payload = "{\"Limit\": 1, \"Filters\": [{\"Values\": [\"unnamed\"], \"Name
\": \"instance-name\"}]}";
string hashedRequestPayload = sha256Hex(payload);
string canonicalRequest = httpRequestMethod + "\n" + canonicalUri + "\n" + canoni
calQueryString + "\n"
+ canonicalHeaders + "\n" + signedHeaders + "\n" + hashedRequestPayload;
cout << canonicalRequest << endl;
cout << "-----------------------" << endl;

// ************* Step 2: Concatenate the string to sign *************
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" + credentialScop
e + "\n" + hashedCanonicalRequest;
cout << stringToSign << endl;
cout << "-----------------------" << endl;

// ************* Step 3: Calculate the signature ***************
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;
cout << "-----------------------" << endl;

// ************* Step 4: Concatenate the Authorization *************
string authorization = algorithm + " " + "Credential=" + SECRET_ID + "/" + creden
tialScope + ", "
+ "SignedHeaders=" + signedHeaders + ", " + "Signature=" + signature;
cout << authorization << endl;
cout << "-----------------------" << endl;

string headers = "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 << headers << endl;
return 0;
};
```

# Signature Failure

The following situational error codes for signature failure may occur. Please resolve the errors accordingly.

| Error Code | Description |
| --- | --- |
| AuthFailure.SignatureExpire | Signature expired. Timestamp and server time cannot differ by more than five minutes. |
| AuthFailure.SecretIdNotFound | The key does not exist. Please go to the console to check whether it is disabled or you copied fewer or more characters. |
| AuthFailure.SignatureFailure | Signature error. It is possible that the signature was calculated incorrectly, the signature does not match the content actually sent, or the SecretKey is incorrect. |
| AuthFailure.TokenFailure | Temporary certificate token error. |
| AuthFailure.InvalidSecretId | Invalid key (not a TencentCloud API key type). |

# Responses

Last updated：2024-09-29 21:28:45

## Response for Successful Requests

For example, when calling CAM API (version: 2017-03-12) to view the status of instances (DescribeInstancesStatus), if the request has succeeded, you may see the response as shown below:

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

- The API will return `Response`, which contains `RequestId`, as long as it processes the request. It does not matter if the request is successful or not.
- RequestId is the unique ID of an API request. Contact us with this ID when an exception occurs.
- Except for the fixed fields, all fields are action-specified. For the definitions of action-specified fields, see the corresponding API documentation. In this example, `TotalCount` and `InstanceStatusSet` are the fields specified by the API `DescribeInstancesStatus`. 0 `TotalCount` means that the requester owns 0 CVM instance so the `InstanceStatusSet` is empty.

## Response for Failed Requests

If the request has failed, you may see the response as shown below:

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

- The presence of the `Error` field indicates that the request has failed. A response for a failed request will include `Error`, `Code` and `Message` fields.
- `Code` is the code of the error that helps you identify the cause and solution. There are two types of error codes so you may find the code in either common error codes or API-specified error codes.
- Message explains the cause of the error. Note that the returned messages are subject to service updates. The information the messages provide may not be up-to-date and should not be the only source of reference.
- RequestId is the unique ID of an API request. Contact us with this ID when an exception occurs.

## Common Error Codes

If there is an Error field in the response, it means that the API call failed. The Code field in Error indicates the error code. The following table lists the common error codes that all actions can return.

| Error Code | Description |
| --- | --- |
| AuthFailure.InvalidSecretId | Invalid key (not a TencentCloud API key type). |
| AuthFailure.MFAFailure | MFA failed. |
| AuthFailure.SecretIdNotFound | The key does not exist. |
| AuthFailure.SignatureExpire | Signature expired. |
| AuthFailure.SignatureFailure | Signature error. |
| AuthFailure.TokenFailure | Token error. |
| AuthFailure.UnauthorizedOperation | The request does not have CAM authorization. |
| DryRunOperation | DryRun Operation. It means that the request would have succeeded, but the DryRun parameter was used. |
| FailedOperation | Operation failed. |
| InternalError | Internal error. |
| InvalidAction | The API does not exist. |
| InvalidParameter | Incorrect parameter. |
| InvalidParameterValue | Invalid parameter value. |
| LimitExceeded | Quota limit exceeded. |
| MissingParameter | A parameter is missing. |

| | |
|---|---|
| NoSuchVersion | The API version does not exist. |
| RequestLimitExceeded | The number of requests exceeds the frequency limit. |
| ResourceInUse | Resource is in use. |
| ResourceInsufficient | Insufficient resource. |
| ResourceNotFound | The resource does not exist. |
| ResourceUnavailable | Resource is unavailable. |
| UnauthorizedOperation | Unauthorized operation. |
| UnknownParameter | Unknown parameter. |
| UnsupportedOperation | Unsupported operation. |
| UnsupportedProtocol | HTTPS request method error. Only GET and POST requests are supported. |
| UnsupportedRegion | API does not support the requested region. |

# Image Face Fusion (Basic) APIs
# FuseFace

Last updated：2024-11-26 11:44:28

## 1. API Description

Domain name for API request: facefusion.intl.tencentcloudapi.com.

This API is used to perform the fusion of a single face, multiple faces, and specified faces with the material template by uploading face images. Users can add logos to generated images. See Fusion Access Guide.

- The signature method in the public parameters must be specified as the V3 version. In other words, set the SignatureMethod parameter to TC3-HMAC-SHA256.

We recommend you to use API Explorer
Try it
API Explorer provides a range of capabilities, including online call, signature authentication, SDK code generation, and API quick search. It enables you to view the request, response, and auto-generated examples.

## 2. Input Parameters

The following request parameter list only provides API request parameters and some common parameters. For the complete common parameter list, see Common Request Parameters.

| Parameter Name | Required | Type | Description |
| --- | --- | --- | --- |
| Action | Yes | String | Common Params. The value used for this API: FuseFace. |
| Version | Yes | String | Common Params. The value used for this API: 2022-09-27. |
| Region | Yes | String | Common Params. For more information, please see the list of regions supported by the product. |
| ProjectId | Yes | String | Activity ID. Check the ID in the Face Fusion console. |
| ModelId | Yes | String | Material ID. Check the ID in the Face Fusion console. |
| RspImgType | Yes | String | Image return method (url or base64). You cannot use both methods at the same time. The URL is valid for 7 days. |

| | | | |
|---|---|---|---|
| MergeInfos.N | Yes | Array of MergeInfo | Face position information on the user face image and material template image. No more than 6 entries. |
| LogoAdd | No | Integer | Switch indicating whether to add a synthesis logo to the fusion result image. Default value: 1.<br>1: add logo<br>0: do not add logo<br>Other values: add logo<br>It is recommended to use an obvious logo to indicate that the result image uses face fusion technology and is synthesized by AI. |
| LogoParam | No | LogoParam | Logo content settings<br>By default, the text "Synthesized by AI" is added to the bottom right corner of the fusion result image. You can also use other texts. |
| FuseParam | No | FuseParam | Fusion parameter. |

# 3. Output Parameters

| Parameter Name | Type | Description |
|---|---|---|
| FusedImage | String | When RspImgType is set to url, return the URL (valid for 7 days). When RspImgType is set to base64, return the Base64 code. |
| RequestId | String | The unique request ID, generated by the server, will be returned for every request (if the request fails to reach the server for other reasons, the request will not obtain a RequestId). RequestId is required for locating a problem. |

# 4. Example

### Example1 Selecting Face for Fusion

#### Input Example

```
POST / HTTP/1.1
Host: facefusion.intl.tencentcloudapi.com
Content-Type: application/json
X-TC-Action: FuseFace
```

```
<Common request parameters>

{
"LogoParam": {
"LogoRect": {
"Y": 0,
"X": 0,
"Height": 0,
"Width": 0
},
"LogoUrl": "test1.jpg"
},
"ProjectId": "at_1603326187690926080",
"LogoAdd": 1,
"RspImgType": "url",
"MergeInfos": [
{
"Url": "test.jpg",
"TemplateFaceID": "mt_1603586676924403712_1"
}
],
"ModelId": "mt_1603586676924403712"
}
```

**Output Example**

```
{
"Response": {
"FusedImage": "result.jpg",
"RequestId": "1a2e88a4-3614-48a0-96b9-d09bf6de2fe4"
}
}
```

# 5. Developer Resources

## SDK

TencentCloud API 3.0 integrates SDKs that support various programming languages to make it easier for you to call APIs.

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

- Tencent Cloud SDK 3.0 for Go
- Tencent Cloud SDK 3.0 for Node.js
- Tencent Cloud SDK 3.0 for .NET
- Tencent Cloud SDK 3.0 for C++

### Command Line Interface

- Tencent Cloud CLI 3.0

## 6. Error Code

The following only lists the error codes related to the API business logic. For other error codes, see Common Error Codes.

| Error Code | Description |
| --- | --- |
| FailedOperation.BalanceInsufficient | Insufficient balance, failed to open, please recharge and open again. |
| FailedOperation.FaceSizeTooSmall | The face was filtered because it was too small. It is recommended that the face size is not less than 34x34 pixels. |
| FailedOperation.FuseMaterialNotAuth | The material has not been reviewed. |
| FailedOperation.FuseMaterialNotExist | The material does not exist. |
| FailedOperation.ImageDecodeFailed | Image decoding failed. |
| FailedOperation.ImageDownloadError | Image download failed. |
| FailedOperation.ImageResolutionTooSmall | The short edge resolution of the image is lower than 64 pixels. |
| FailedOperation.ImageSizeExceed | The image after Base64 encoding exceeds in size. |
| FailedOperation.ImageSizeInvalid | The image size is too large or too small and does not meet algorithm requirements. |
| FailedOperation.InnerError | Internal service error. |
| FailedOperation.NoFaceDetected | The face cannot be detected because the face box is too small. |
| FailedOperation.ParameterValueError | Parameter or value is invalid. |

| | |
|---|---|
| FailedOperation.RequestTimeout | The backend service timed out. |
| FailedOperation.RpcFail | RPC request failed, typically due to algorithm service malfunction. |
| FailedOperation.TemplateFaceIDNotExist | The material face ID does not exist. |
| FailedOperation.UnKnowError | Internal error. |
| FailedOperation.Unknown | Unknown error. |
| InvalidParameterValue.ActivityIdNotFound | Activity ID is not found. |
| InvalidParameterValue.FaceRectParameterValueError | Face box parameters are invalid, or the face box is too small. |
| InvalidParameterValue.MaterialIdNotFound | No material ID is found. |
| InvalidParameterValue.UrlIllegal | The URL format is invalid. |
| RequestLimitExceeded | The number of requests exceeded the rate limit. |
| ResourceInsufficient | Resources are insufficient. |
| ResourceNotFound | The resource does not exist. |
| ResourceUnavailable.Freeze | The account has been frozen. |
| ResourceUnavailable.InArrears | The account is in arrears. |
| ResourceUnavailable.IsOpening | The service is being opened, please wait. |
| ResourceUnavailable.NotExist | The billing status is unknown. Check whether the service has been activated in the console. |
| ResourceUnavailable.Recover | The resource has been possessed. |
| ResourceUnavailable.StopUsing | Services for the account has been stopped. |

# Video Face Fusion (Single Face) APIs SubmitVideoFaceFusionJob

Last updated：2024-11-26 11:44:27

## 1. API Description

Domain name for API request: facefusion.intl.tencentcloudapi.com.

This API is used to submit asynchronous processing tasks of video face fusion. After a task is submitted, the Job ID, estimated completion time, and current queue length will be returned.

We recommend you to use API Explorer

Try it

API Explorer provides a range of capabilities, including online call, signature authentication, SDK code generation, and API quick search. It enables you to view the request, response, and auto-generated examples.

## 2. Input Parameters

The following request parameter list only provides API request parameters and some common parameters. For the complete common parameter list, see Common Request Parameters.

| Parameter Name | Required | Type | Description |
| --- | --- | --- | --- |
| Action | Yes | String | Common Params. The value used for this API: SubmitVideoFaceFusionJob. |
| Version | Yes | String | Common Params. The value used for this API: 2022-09-27. |
| Region | Yes | String | Common Params. For more information, please see the list of regions supported by the product. |
| ProjectId | Yes | String | Activity ID. Check it in the video face fusion console. |
| ModelId | Yes | String | Material ID. Check it in the video face fusion console. |
| MergeInfos.N | Yes | Array of MergeInfo | Face position information on the user face image and material template image. Only one entry is allowed. |
| CelebrityIdentify | No | Integer | 0: inappropriate content recognition not required; 1: inappropriate content recognition required. Default value: 0. |

| | | | |
|---|---|---|---|
| | | | Note: Once the inappropriate content recognition service is enabled, you need to decide whether to adjust your business logic based on the returned results. For example, you need to replace the image if the system informs you that the image does not meet the requirements.<br>**Note: This field will be deprecated later due to business adjustments. It is not recommended for use.** |
| LogoParam | No | LogoParam | Video watermark logo parameter |
| UserDesignatedUrl | No | String | COS pre-signed URL (PUT method). If this parameter is specified, the video after fusion will be uploaded to this URL.<br>**Note: If upload to this URL fails, the video will be uploaded to the default address of Tencent Cloud.** |
| UserIp | No | String | User IP address |
| MetaData.N | No | Array of MetaData | Video metadata field |

## 3. Output Parameters

| Parameter Name | Type | Description |
|---|---|---|
| JobId | String | Job ID of the video face fusion task |
| EstimatedProcessTime | Float | Estimated processing time of the video face fusion task, in seconds |
| JobQueueLength | Integer | Estimated processing time of the video face fusion task, in seconds |
| ReviewResultSet | Array of FuseFaceReviewResult | Inappropriate content recognition result. The element order of this array is the same as that of mergeinfo in the request, with a one-to-one relationship.<br>Note: This field may return null, indicating that no valid values can be obtained. |
| RequestId | String | The unique request ID, generated by the server, will be returned for every request (if the request fails to reach the server for other reasons, the request will not obtain a RequestId). RequestId is required for locating a problem. |

# 4. Example

## Example1 Submitting Video Face Fusion Tasks

This example shows you how to submit video face fusion tasks.

### Input Example

```
POST / HTTP/1.1
Host: facefusion.intl.tencentcloudapi.com
Content-Type: application/json
X-TC-Action: SubmitVideoFaceFusionJob
<Common request parameters>

{
"ProjectId": "100646",
"MergeInfos": [
{
"Url": "http://i2.sinaimg.cn/ty/nba/2015-07-05/U10236P6T12D7648505F44DT20150705114547.jpg"
}
],
"ModelId": "qc_100646_154021_9"
}
```

### Output Example

```
{
"Response": {
"JobId": "C0a5EXaNR7JzGvlg",
"EstimatedProcessTime": 30,
"JobQueueLength": 1,
"ReviewResultSet": [
{
"Category": "Politics",
"Code": "0",
"CodeDescription": "OK",
"Suggestion": "PASS",
"Confidence": 30,
"DetailSet": [
{
"Field": "",
"Label": "Ding Junhui",
"Confidence": 30,
"Suggestion": "PASS"
```

```
}
]
}
],
"RequestId": "83ecff39-2e4a-41d5-8562-1f8898326565"
}
}
```


# 5. Developer Resources

### SDK

TencentCloud API 3.0 integrates SDKs that support various programming languages to make it easier for you to call APIs.

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

### Command Line Interface

- Tencent Cloud CLI 3.0

# 6. Error Code

The following only lists the error codes related to the API business logic. For other error codes, see Common Error Codes.

| Error Code | Description |
| --- | --- |
| FailedOperation.BalanceInsufficient | Insufficient balance, failed to open, please recharge and open again. |
| FailedOperation.FaceIdNotInVideo | The face corresponding to the specified ID does not exist in the video. |
| FailedOperation.FaceSizeTooSmall | The face was filtered because it was too small. It is |

| | |
|---|---|
| | recommended that the face size is not less than 34x34 pixels. |
| FailedOperation.FuseMaterialNotAuth | The material has not been reviewed. |
| FailedOperation.FuseMaterialNotAvailable | The material in this state cannot be used. |
| FailedOperation.FuseMaterialNotExist | The material does not exist. |
| FailedOperation.ImageDecodeFailed | Image decoding failed. |
| FailedOperation.ImageDownloadError | Image download failed. |
| FailedOperation.ImageResolutionExceed | The image size is too large. It is recommended to resize the image to below 2,000x2,000 pixels. |
| FailedOperation.ImageResolutionTooSmall | The short edge resolution of the image is lower than 64 pixels. |
| FailedOperation.ImageSizeExceed | The image after Base64 encoding exceeds in size. |
| FailedOperation.ImageSizeInvalid | The image size is too large or too small and does not meet algorithm requirements. |
| FailedOperation.InnerError | Internal service error. |
| FailedOperation.NoFaceDetected | The face cannot be detected because the face box is too small. |
| FailedOperation.ParameterValueError | Parameter or value is invalid. |
| FailedOperation.ProjectNotAuth | The authorization fee is not paid for the activity, or the activity has been disabled. |
| FailedOperation.RequestTimeout | The backend service timed out. |
| FailedOperation.RpcFail | RPC request failed, typically due to algorithm service malfunction. |
| FailedOperation.UnKnowError | Internal error. |
| FailedOperation.Unknown | Unknown error. |
| InvalidParameterValue.FaceRectParameterValueError | Face box parameters are invalid, or the face box is too small. |
| InvalidParameterValue.UrlIllegal | The URL format is invalid. |
| RequestLimitExceeded | The number of requests exceeded the rate limit. |

| | |
|---|---|
| ResourceInsufficient | Resources are insufficient. |
| ResourceNotFound | The resource does not exist. |
| ResourceUnavailable.Freeze | The account has been frozen. |
| ResourceUnavailable.InArrears | The account is in arrears. |
| ResourceUnavailable.IsOpening | The service is being opened, please wait. |
| ResourceUnavailable.NotExist | The billing status is unknown. Check whether the service has been activated in the console. |
| ResourceUnavailable.Recover | The resource has been possessed. |
| ResourceUnavailable.StopUsing | Services for the account has been stopped. |

# QueryVideoFaceFusionJob

Last updated：2024-11-26 11:44:28

## 1. API Description

Domain name for API request: facefusion.intl.tencentcloudapi.com.

This API is used to query the progress and status of video face fusion tasks by Job ID.

We recommend you to use API Explorer

Try it

API Explorer provides a range of capabilities, including online call, signature authentication, SDK code generation, and API quick search. It enables you to view the request, response, and auto-generated examples.

## 2. Input Parameters

The following request parameter list only provides API request parameters and some common parameters. For the complete common parameter list, see Common Request Parameters.

| Parameter Name | Required | Type | Description |
| --- | --- | --- | --- |
| Action | Yes | String | Common Params. The value used for this API: QueryVideoFaceFusionJob. |
| Version | Yes | String | Common Params. The value used for this API: 2022-09-27. |
| Region | Yes | String | Common Params. For more information, please see the list of regions supported by the product. |
| JobId | Yes | String | Job ID of the video face fusion task |

## 3. Output Parameters

| Parameter Name | Type | Description |
| --- | --- | --- |
| JobStatus | String | Current task status: queuing, processing, processing failed, or processing completed |
| | | |

| | | |
|---|---|---|
| VideoFaceFusionOutput | VideoFaceFusionOutput | Video face fusion result<br>Note: This field may return null, indicating that no valid values can be obtained. |
| JobStatusCode | Integer | Task status code. 1: queuing; 3: processing; 5: processing failed; 7: processing completed.<br>Note: This field may return null, indicating that no valid values can be obtained. |
| JobErrorCode | String | Task failure error code<br>Note: This field may return null, indicating that no valid values can be obtained. |
| JobErrorMsg | String | Task failure error message<br>Note: This field may return null, indicating that no valid values can be obtained. |
| RequestId | String | The unique request ID, generated by the server, will be returned for every request (if the request fails to reach the server for other reasons, the request will not obtain a RequestId). RequestId is required for locating a problem. |

# 4. Example

## Example1 Querying Video Face Fusion Tasks

This example shows you how to query video face fusion tasks.

### Input Example

```
https://facefusion.intl.tencentcloudapi.com/?Action=QueryVideoFaceFusionJob
&JobId=C0a5EXaNR7JzGvlg
&<Common request parameters>
```

### Output Example

```
{
"Response": {
"JobStatusCode": 7,
"JobStatus": "Processing completed",
"VideoFaceFusionOutput": {
"VideoUrl": "http://bda-video-bodyseg-1254418846.cos.ap-guangzhou.myqcloud.com/video_fusion_test/1.0/251006455/20210119164402_83ecff39-2e4a-41d5-8562-1f8898326565
```

```
_qc_300314_789050_75_1611045834157_result.mp4",
"VideoMD5": "3AA00F9A2914DF3F2268628C45C4E4CE",
"Width": 720,
"Height": 1280,
"FPS": 25,
"DurationInSec": 15.079999923706,
"Frame": 375
},
"JobErrorCode": "",
"JobErrorMsg": "",
"RequestId": "ef13456e-8174-4418-8e51-a724257c9a3a"
}
}
```


# 5. Developer Resources

### SDK

TencentCloud API 3.0 integrates SDKs that support various programming languages to make it easier for you to call APIs.

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

### Command Line Interface

- Tencent Cloud CLI 3.0

# 6. Error Code

The following only lists the error codes related to the API business logic. For other error codes, see Common Error Codes.

| Error Code | Description |
| --- | --- |
| FailedOperation.InnerError | Internal service error. |

| | |
|---|---|
| FailedOperation.JobHasBeenCanceled | The task has been canceled. Please submit the task again. |
| FailedOperation.JobNotExist | The task does not exist. |
| FailedOperation.NoFaceDetected | The face cannot be detected because the face box is too small. |
| FailedOperation.ParameterValueError | Parameter or value is invalid. |
| FailedOperation.RequestTimeout | The backend service timed out. |
| FailedOperation.RpcFail | RPC request failed, typically due to algorithm service malfunction. |
| FailedOperation.UnKnowError | Internal error. |
| FailedOperation.Unknown | Unknown error. |

# Data Types

Last updated：2024-11-06 15:17:41

## FaceRect

Face box information

Used by actions: FuseFace, SubmitVideoFaceFusionJob.

| Name | Type | Required | Description |
| --- | --- | --- | --- |
| X | Integer | Yes | Top-left X-axis coordinate of the face box |
| Y | Integer | Yes | Top-left Y-axis coordinate of the face box |
| Width | Integer | Yes | Face box width |
| Height | Integer | Yes | Face box height |

## FuseFaceReviewDetail

Used by actions: SubmitVideoFaceFusionJob.

| Name | Type | Description |
| --- | --- | --- |
| Field | String | |
| Label | String | |
| Confidence | Float | |
| Suggestion | String | |

## FuseFaceReviewResult

Used by actions: SubmitVideoFaceFusionJob.

| Name | Type | Description |
| --- | --- | --- |
| Category | String | |

| | | |
|---|---|---|
| Code | String | |
| CodeDescription | String | |
| Confidence | Float | |
| Suggestion | String | |
| DetailSet | Array of FuseFaceReviewDetail | |

# FuseParam

Fusion parameter

Used by actions: FuseFace.

| Name | Type | Required | Description |
|---|---|---|---|
| ImageCodecParam | ImageCodecParam | No | Image encoding parameter |

# ImageCodecParam

Image encoding parameter

Used by actions: FuseFace.

| Name | Type | Required | Description |
|---|---|---|---|
| MetaData | Array of MetaData | No | Metadata. The number of metadata entries cannot exceed 1. |

# LogoParam

Logo parameter

Used by actions: FuseFace, SubmitVideoFaceFusionJob.

| Name | Type | Required | Description |
|---|---|---|---|
| LogoRect | FaceRect | Yes | Coordinates of the logo image in the fusion result image. The logo image will be stretched according to the coordinates. |
| LogoUrl | String | No | Logo image URL |

| | | | |
|---|---|---|---|
| | | | ●Either the Base64 code or URL must be provided. If both are provided, URL prevails.<br>●Supported image format: JPG or PNG |
| LogoImage | String | No | Logo image Base64 code<br>●Either the Base64 code or URL must be provided. If both are provided, URL prevails.<br>●Supported image format: JPG or PNG |

## MergeInfo

Face position information on the face image and material template image for fusion

Used by actions: FuseFace, SubmitVideoFaceFusionJob.

| Name | Type | Required | Description |
|---|---|---|---|
| Image | String | No | Enter the image Base64 code.<br>●Either the Base64 code or URL must be provided. If both are provided, URL prevails.<br>●Material image limitation: face size in the image greater than 34×34 pixels; image size greater than 64×64 pixels. (After encoding, the image size may increase by about 30%. It is recommended to control the image size reasonably.)<br>●Supported image format: JPG or PNG |
| Url | String | No | Enter the image URL.<br>●Either the Base64 code or URL must be provided. If both are provided, URL prevails.<br>●Material image limitation: face size in the image greater than 34×34 pixels; image size greater than 64×64 pixels. (After encoding, the image size may increase by about 30%. It is recommended to control the image size reasonably.)<br>●Supported image format: JPG or PNG |
| InputImageFaceRect | FaceRect | No | Face position information (face box) on the uploaded image Width and height are no less than 30. |
| TemplateFaceID | String | No | Material face ID. If this parameter is left blank, the largest face is used by default. |
| TemplateFaceRect | FaceRect | No | Face position information (face box) on the template. If this parameter is left blank, the largest face is used by default. |

| | | | |
|---|---|---|---|
| | | | This parameter applies to scenes where custom material templates are used for fusion.<br>Width and height are no less than 30. |

## MetaData

Metadata structure, in key/value format

Used by actions: FuseFace, SubmitVideoFaceFusionJob.

| Name | Type | Required | Description |
|---|---|---|---|
| MetaKey | String | Yes | Metadata key |
| MetaValue | String | Yes | Metadata value |

## VideoFaceFusionOutput

Returned video face fusion result

Used by actions: QueryVideoFaceFusionJob.

| Name | Type | Description |
|---|---|---|
| VideoUrl | String | URL of the video output after video face fusion |
| VideoMD5 | String | MD5 value of the video output after video face fusion, which is used for verification |
| Width | Integer | Video width |
| Height | Integer | Video height |
| FPS | Integer | Frames per second |
| DurationInSec | Float | Video duration, in seconds |
| Frame | Integer | Number of frames |

# Error Codes

Last updated：2024-11-06 15:17:41

## Feature Description

If there is an Error field in the response, it means that the API call failed. For example:

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

Code in Error indicates the error code, and Message indicates the specific information of the error.

## Error Code List

### Common Error Codes

| Error Code | Description |
| --- | --- |
| ActionOffline | This API has been deprecated. |
| AuthFailure.InvalidAuthorization | `Authorization` in the request header is invalid. |
| AuthFailure.InvalidSecretId | Invalid key (not a TencentCloud API key type). |
| AuthFailure.MFAFailure | MFA failed. |
| AuthFailure.SecretIdNotFound | Key does not exist. Check if the key has been deleted or disabled in the console, and if not, check if the key is correctly entered. Note that whitespaces should not exist before or after the key. |
| AuthFailure.SignatureExpire | Signature expired. Timestamp and server time cannot differ by more than five minutes. Please |

| | |
|---|---|
| | ensure your current local time matches the standard time. |
| AuthFailure.SignatureFailure | Invalid signature. Signature calculation error. Please ensure you've followed the signature calculation process described in the Signature API documentation. |
| AuthFailure.TokenFailure | Token error. |
| AuthFailure.UnauthorizedOperation | The request is not authorized. For more information, see the CAM documentation. |
| DryRunOperation | DryRun Operation. It means that the request would have succeeded, but the DryRun parameter was used. |
| FailedOperation | Operation failed. |
| InternalError | Internal error. |
| InvalidAction | The API does not exist. |
| InvalidParameter | Incorrect parameter. |
| InvalidParameterValue | Invalid parameter value. |
| InvalidRequest | The multipart format of the request body is incorrect. |
| IpInBlacklist | Your IP is in uin IP blacklist. |
| IpNotInWhitelist | Your IP is not in uin IP whitelist. |
| LimitExceeded | Quota limit exceeded. |
| MissingParameter | A parameter is missing. |
| NoSuchProduct | The product does not exist. |
| NoSuchVersion | The API version does not exist. |
| RequestLimitExceeded | The number of requests exceeds the frequency limit. |
| RequestLimitExceeded.GlobalRegionUinLimitExceeded | Uin exceeds the frequency limit. |
| RequestLimitExceeded.IPLimitExceeded | The number of ip requests exceeds the frequency limit. |
| RequestLimitExceeded.UinLimitExceeded | The number of uin requests exceeds the frequency |

| | |
|---|---|
| | limit. |
| RequestSizeLimitExceeded | The request size exceeds the upper limit. |
| ResourceInUse | Resource is in use. |
| ResourceInsufficient | Insufficient resource. |
| ResourceNotFound | The resource does not exist. |
| ResourceUnavailable | Resource is unavailable. |
| ResponseSizeLimitExceeded | The response size exceeds the upper limit. |
| ServiceUnavailable | Service is unavailable now. |
| UnauthorizedOperation | Unauthorized operation. |
| UnknownParameter | Unknown parameter. |
| UnsupportedOperation | Unsupported operation. |
| UnsupportedProtocol | HTTP(S) request protocol error; only GET and POST requests are supported. |
| UnsupportedRegion | API does not support the requested region. |

## Service Error Codes

| Error Code | Description |
|---|---|
| FailedOperation.BalanceInsufficient | Insufficient balance, failed to open, please recharge and open again. |
| FailedOperation.FaceIdNotInVideo | The face corresponding to the specified ID does not exist in the video. |
| FailedOperation.FaceSizeTooSmall | The face was filtered because it was too small. It is recommended that the face size is not less than 34x34 pixels. |
| FailedOperation.FuseMaterialNotAuth | The material has not been reviewed. |
| FailedOperation.FuseMaterialNotAvailable | The material in this state cannot be used. |
| FailedOperation.FuseMaterialNotExist | The material does not exist. |
| FailedOperation.ImageDecodeFailed | Image decoding failed. |

| | |
|---|---|
| FailedOperation.ImageDownloadError | Image download failed. |
| FailedOperation.ImageResolutionExceed | The image size is too large. It is recommended to resize the image to below 2,000x2,000 pixels. |
| FailedOperation.ImageResolutionTooSmall | The short edge resolution of the image is lower than 64 pixels. |
| FailedOperation.ImageSizeExceed | The image after Base64 encoding exceeds in size. |
| FailedOperation.ImageSizeInvalid | The image size is too large or too small and does not meet algorithm requirements. |
| FailedOperation.InnerError | Internal service error. |
| FailedOperation.JobHasBeenCanceled | The task has been canceled. Please submit the task again. |
| FailedOperation.JobNotExist | The task does not exist. |
| FailedOperation.NoFaceDetected | The face cannot be detected because the face box is too small. |
| FailedOperation.ParameterValueError | Parameter or value is invalid. |
| FailedOperation.ProjectNotAuth | The authorization fee is not paid for the activity, or the activity has been disabled. |
| FailedOperation.RequestTimeout | The backend service timed out. |
| FailedOperation.RpcFail | RPC request failed, typically due to algorithm service malfunction. |
| FailedOperation.TemplateFaceIDNotExist | The material face ID does not exist. |
| FailedOperation.UnKnowError | Internal error. |
| FailedOperation.Unknown | Unknown error. |
| InvalidParameterValue.ActivityIdNotFound | Activity ID is not found. |
| InvalidParameterValue.FaceRectParameterValueError | Face box parameters are invalid, or the face box is too small. |
| InvalidParameterValue.MaterialIdNotFound | No material ID is found. |
| InvalidParameterValue.UrlIllegal | The URL format is invalid. |
| ResourceUnavailable.Freeze | The account has been frozen. |

| | |
|---|---|
| ResourceUnavailable.InArrears | The account is in arrears. |
| ResourceUnavailable.IsOpening | The service is being opened, please wait. |
| ResourceUnavailable.NotExist | The billing status is unknown. Check whether the service has been activated in the console. |
| ResourceUnavailable.Recover | The resource has been possessed. |
| ResourceUnavailable.StopUsing | Services for the account has been stopped. |