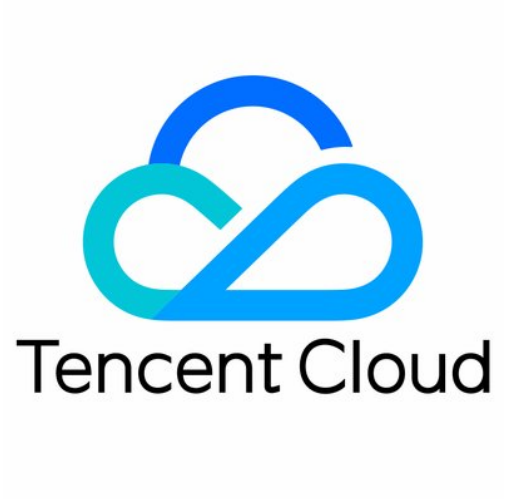
Cloud Message Queue

SDK

Product Documentation



Contents

SDK

SDK Release Logs

Last updated：2022-03-29 16:30:56

CMQ SDK for HTTP 1.0.7 (March 25, 2020)

- Changed `artifactId` to `cmq-http-client` on the new version.
- Fixed the bug where messages exceptionally got lost.
- Removed unused clients and fixed issues #6 and #7.

Code repository address >>

CMQ SDK for TCP 1.1.1 (February 12, 2020)

New features

- Added stream management APIs, such as those for queue creation and subscription creation.
- Completed relevant unit testings, which are not enabled during construction by default.
- Made the `pom` file compatible with Maven 3.0+.

Bug fixes

- Fixed the issue where transaction messages exceptionally got lost.
- Fixed the issue where the timeout period in the configuration did not take effect for TCP.
- Fixed the exception of "repeated authentication" in concurrence scenarios.

Code repository address >>

CMQ SDK for HTTP 1.0.6 (November 12, 2019)

New features

- SDK configuration is objectified and parameters such as timeout period and whether to print slow operation logs are added. For more information, please see the `CmqConfig` object.
- The new message sending and receiving APIs feature 6 times higher performance.
- Logs for slow operations and exceptions can be printed automatically (which requires a log file to be configured).
- The return of `requestId` is added for the new sending API for problem tracking.

Bug fixes

- Fixed the occasional exception of JSON serialization.
- Fixed other code specification issues.

- Fixed log configuration issues.
- The SDK uses SLF4J to print logs, but there is no specific log configuration file. You can add a configuration file of logging frameworks such as Log4j or Logback to `classpath` .

Usage recommendations

- Receive messages by using the new (batch) `receiveMessage` API (long polling wait time is subject to queue settings).
- Send messages by using the new (batch) `send` API (which can return the `requestId`).

Code repository address >>

CMQ SDK 1.05 (June 28, 2019)

CMQ now supports SDK calls based on the TCP protocol, which feature lower computing resource usage, higher client thread security, transfer efficiency, and feature diversity, and better user experience.

CMQ SDK 1.0.4 (April 7, 2017)

- The CMQ SDK now supports SHA256 signature algorithm. You can call `setSignMethod` during account initialization to set the signature algorithm (for specific algorithm names, please see the code description).
- Known bugs are fixed.

CMQ SDK 1.0.3 (March 13, 2017)

CMQ now supports message rewind, message delay, and subscription routing.

CMQ SDK 1.0.2 (December 1, 2016)

- The SDK supports topic and subscription modes.
- CMake is used for project management in the SDK for C++.
- Maven is used for project management in the SDK for Java.

CMQ SDK 1.0.1 (October 12, 2016)

- Optimized the user experience in case of client timeout.
- Fixed the bug of authentication failure in the SDK for PHP.
- Fixed the bug of message sending in PHP.

CMQ SDK 1.0.0 (September 7, 2016)

- The versions for C++, Java, Python, and PHP are released.
- The SDK encapsulates APIs for operations such as sending, receiving, and deleting messages.

HTTP SDK

Last updated：2020-06-08 16:09:10

⚠ Note：

This is a legacy API which has been hidden and will no longer be updated. We recommend using the new CMQ API 3.0 which is standardized and faster.

Overview

Tencent Cloud message queue CMQ currently supports Java, Python, PHP, and C + + SDK, more languages will be supported in the future. Developers are welcome to develop more language versions of SDK according to API instructions.
Since it takes about 1 second for Assign resources and resources to be released, the current message queuing SDK has a 1 second delay when creating and deleting queues / topic. It is recommended to increase the interval between creation and deletion in the program to ensure a successful call.

- To ensure the security of data transmission and provide you with more reliable services, the Tencent Cloud team will stop treating topic and Access as the public network HTTP of the queue at 23:59:59 on January 31st, 2019. It is recommended that you change the domain name of the public network API request to HTTPS.
- You can modify the domain name requested by the public network interface as follows:
  Determine whether the Protocol used in the connected domain name is HTTP,. If so, replace it with HTTPS.
  For example: `endpoint=http://cmq-topic-gz.api.qcloud.com` The access method is HTTP Protocol Access Guangzhou topic public network domain name, modified as follows: `endpoint=https://cmq-topic-gz.api.qcloud.com` .

Download URL

The GitHub addresses of different language versions of SDK are as follows:

GitHub address

- Java SDK
- Python SDK (default is Python2 SDK,. You can switch to the Python3 branch to view the Python3 SDK)
- PHP SDK
- C+ +SDK

Notes

Before using SDK, you should at least get the SecretID 、 SecretKey And endpoint (that is, which region the request is sent to, private network or public network). Endpoint is described below.

Queue model

Please refer to the following instructions to replace {$region} in the domain name with the corresponding region:

- Domain name requested by public network API: `https://cmq-queue-{$region}.api.qcloud.com`
- Domain name requested by private network API: `http://cmq-queue-{$region}.api.tencentyun.com`

Topic model

Please refer to the following instructions to replace {$region} in the domain name with the corresponding region:

- Domain name requested by public network API: `https://cmq-topic-{$region}.api.qcloud.com`
- Domain name requested by private network API: `http://cmq-topic-{$region}.api.tencentyun.com`

If the business process is also deployed on Tencent Cloud's CVM CVM, it is strongly recommended to use Intra-region 's private network endpoint. For example, CVM, CVM of Tencent Cloud in Beijing, is recommended to use `http://cmq-queue-bj.api.tencentyun.com` . Because:

- Intra-region, private network and Latency are even lower.
- At present, the message queue for the public network downstream Traffic is to collect Traffic cost, with private network can save this part of cost.

Description

{$region} need to replace: gz (Guangzhou), sh (Shanghai), bj (Beijing), shjr (Shanghai Finance), szjr (Shenzhen Finance), hk (Hong Kong), cd (Chengdu), ca (North American), usw (Maxi), sg (Singapore with a specific region. The region value in the common parameters should be consistent with the region value of the domain name. If there is any inconsistency, the request will be sent to the region specified by the domain name region based on the region value of the domain name.

Demo project use

Prepare the Demo environment

1. **Install IDE**

   You can install IntelliJ IDEA or Eclipse,. This article uses IntelliJ IDEA as an example.

   Please Download the IntelliJ IDEA Ultimate version And refer to the IntelliJ IDEA instructions for installation

2. **Download the Demo project**

   Please Download the Demo project for CMQ-HTTP When you unzip it locally, you can see the new local cmq-java-sdk-master folder.

Configure the Demo project

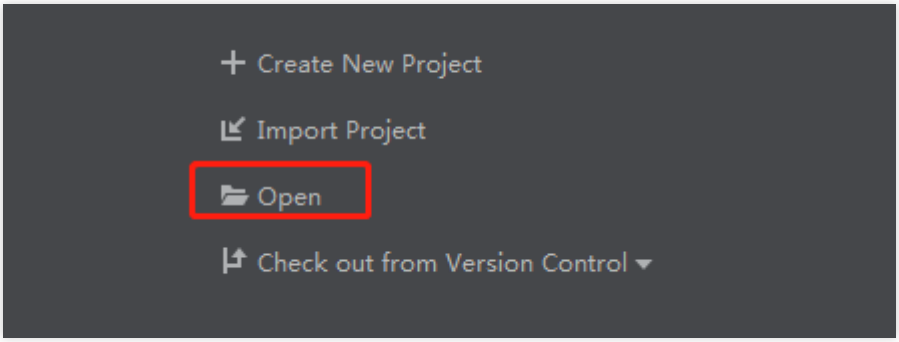
1. **Create Resource**

   You need to create the required message queuing resources in the console, including CMQ queue name, SecretID, SecretKey.

   For the specific creation process, please refer to Queue model Getting Started and Topic model Getting Started .

2. **Import Demo project files**

   Open the folder in the Start up interface of IDEA.



   After opening the folder, the Demo project file is stored in the `/src/main/java/com/qcloud/cmq/example` Under the folder.

3. **Configure Demo parameters**

   Modify file request address and key peer. Take Producer as an example, the configuration is as follows:

```
String secretId="Acquired SecretID";
String secretKey="Acquired SecretKey";
String endpoint = "https://cmq-topic-{$region}.api.qcloud.com";
String queueName = "test";
```

Select "Create queue" and "use existing queue":

```
//Create a new queue and set properties
QueueMeta meta = new QueueMeta();
meta.pollingWaitSeconds = 10;
```

```
meta.visibilityTimeout = 10;
meta.maxMsgSize = 1048576;
meta.msgRetentionSeconds = 345600;
Queue queue = account.createQueue(queueName,meta);

//Use the console existing queue
Queue queue = account.getQueue(queueName);
```

This Demo uses the existing queue of the current account. If you need to select the Create queue, change the queueName to the name of the queue to be created, and then find the Create queue-related code to cancel the comments.

The comment code is a common operation, the "delete" operation should be used with caution, and the configuration in other classes is referred to the Producer class.

Run Demo

Send and receive messages using queue model

First run the Producer class to send the message, and then run the Consumer class to accept the message.

Example of sending message code:

```
String msg = "hello!";
String msgId = queue.sendMessage(msg);
System.out.println("==> send success! msg_id:" + msgId);
```

Example of receiving message code:

```
Message msg = queue.receiveMessage(10);
```

Send and receive messages using topic model

Run the TopicDemo class, and topic's model requests domain name reference. Topic model request domain name .

Publish message example:

```
String msg = "hello!";
String msgId = topic.publishMessage(msg);
```

Example of processing messages:

```
String queueName = "test";
String subscriptionName = "sub-test";
String Endpoint = queueName;
String Protocol = "queue";
account.createSubscribe(topicName,subscriptionName, Endpoint, Protocol);
```

When creating a subscriber, Enter uses a queue to process messages.

SDK for TCP

Last updated：2020-09-10 14:55:43

CMQ currently allows SDK calls based on the TCP protocol for sending and receiving general messages, transaction messages, delayed messages, and async messages. Among them, transaction messages can only be implemented through TCP.
The TCP protocol currently supports public network access and private network access from a CVM instance in a VPC in the same region as the CMQ instance, but does not support the private network access over the classic network.
This document describes how to use the SDK for TCP and install, download, configure, and run the demo to help you quickly build CMQ test projects.

TCP Protocol Advantages

- **Fewer computing resources**
  HTTP authenticates requests and requires a signature for each request, while TCP authenticates links; therefore, it is sufficient to authenticate a link during its establishment, which saves client computing resources.
- **Safer client threads**
  The TCP client is thread-safe, and multiple threads can use the same link, which saves link resources.
- **Higher transfer efficiency**
  TCP increases the proportion of valid data during transfer. For the same client, it has higher throughput and QPS and greater transfer efficiency than HTTP.
- **Better user experience**
  TCP supports async APIs and callbacks.
- **More features**
  TCP supports the latest CMQ feature of distributed message transaction.

Demo Usage

Preparing demo environment

1. **Install IDE**
   You can install IntelliJ IDEA or Eclipse. This document uses IntelliJ IDEA as an example.
   Please download IntelliJ IDEA Ultimate and install it accordingly.
2. **Download the demo**
   Please download the CMQ demo to your local system and decompress it. You will see the `cmq-java-tcp-sdk-`

`master` folder then.

Configuring demo

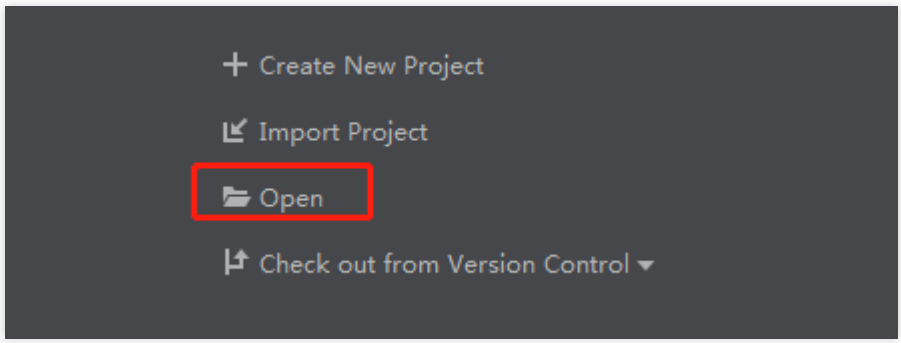
1. **Create a resource**

   You need to create the required CMQ resource in the console and get the CMQ queue name, `SecretID` , and `SecretKey` .

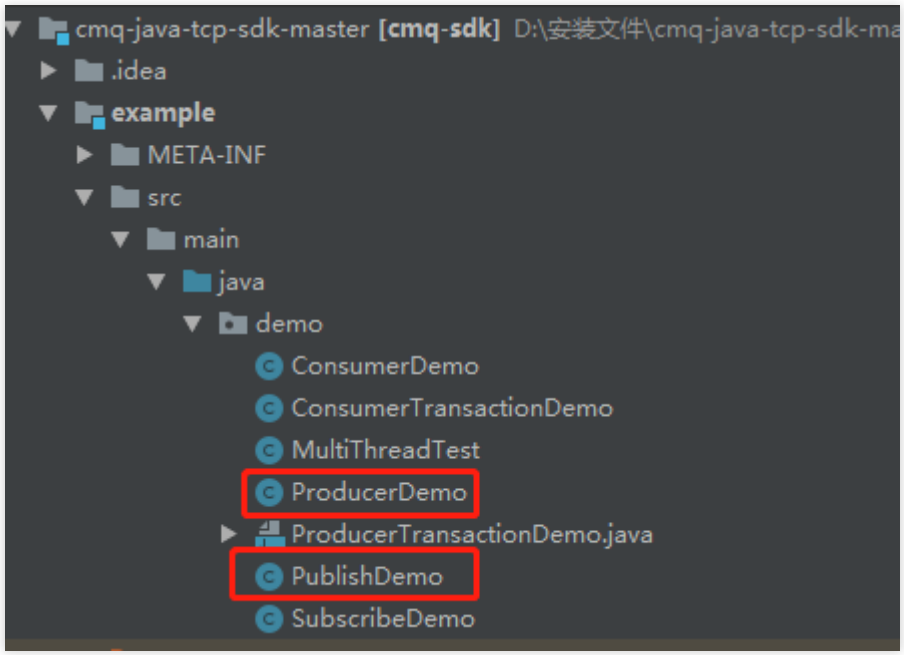
   For detailed directions, please see Queue Model and Topic Model.

2. **Import demo files**

   Open the folder on the startup page of IDEA.



   After the folder is opened, the file hierarchy is as follows, and the demo files are stored in the `demo` folder.



3. **Configure demo parameters**

   Modify the file NameServer address, key pair, and queue name. For the NameServer address, please see the NameServer Table.

   Taking ProducerDemo as an example, the configuration is as follows:

```
producer.setNameServerAddress("Corresponding NameServer");
producer.setSecretID("Obtained SecretID")
producer.setSecretKey("Obtained SecretKey")
String queue = "Name of the created queue"
```

The details are as follows:

```
public class ProducerDemo {
    public static void main(String args[]) {

        Producer producer = new Producer();

        producer.setNameServerAddress("http://cmq-nameserver-bj.tencentcloudapi.com");

        producer.setSecretId("AKIDX1LbH2DNo8MWmOdx                DVWc");

        producer.setSecretKey("F2jJjAWr5A5v8D8sT                 ");

        producer.setSignMethod(ClientConfig.SIGN_METHOD_SHA256);

        producer.setRetryTimesWhenSendFailed(3);

        producer.setRequestTimeoutMS(8000);


        String queue = "subuin";
        try {
```

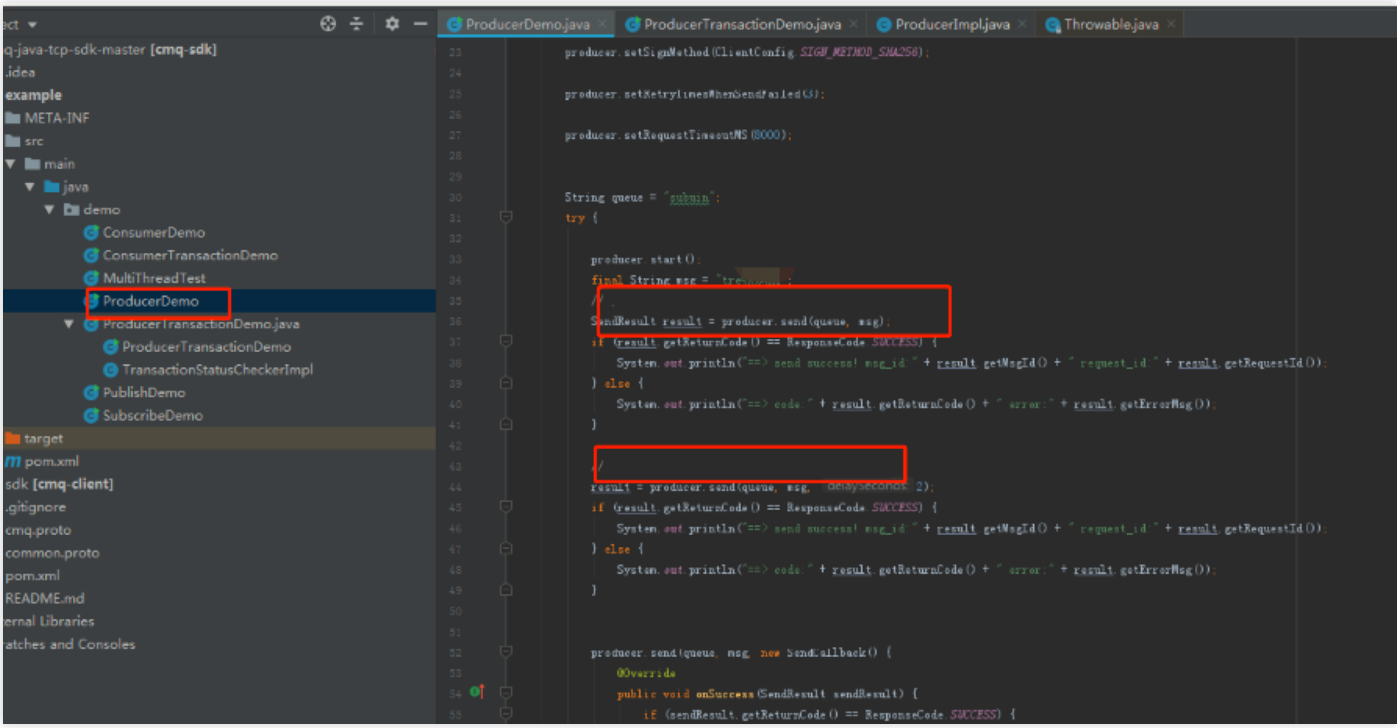
Running demo

Sending and receiving messages with queue model

1. Configure demo parameters.
2. After the `ProducerDemo` file is successfully executed, the log will be displayed as follows:

```
2019-06-26 17:26:32,448 INFO  NettyClient.java(666) : createChannel: begin to connect remote host[140.143.52.21:12000] asynchronously
2019-06-26 17:26:32,470 INFO  NettyClient.java(760) : NETTY CLIENT PIPELINE: CONNECT  UNKNOWN => 140.143.52.21:12000
==> send success! msg_id:4222124650659902 request_id:1
==> send success! msg_id:4222124650659903 request_id:3
==> send success! msg_id:4222124650659904 request_id:4
==> send success! msg_id:[4222124650659905, 4222124650659906] request_id:3
==> send success! msg_id:[4222124650659907, 4222124650659908] request_id:6
2019-06-26 17:26:37,890 INFO  NettyClient.java(573) : closeChannel: begin close the channel[140.143.52.21:12000] Found: true
2019-06-26 17:26:37,890 INFO  NettyClient.java(586) : closeChannel: the channel[140.143.52.21:12000] was removed from channel table
2019-06-26 17:26:37,891 INFO  NettyClient.java(775) : NETTY CLIENT PIPELINE: CLOSE 140.143.52.21:12000
2019-06-26 17:26:37,892 INFO  MQClientInstance.java(108) : the client instance [10.66.130.82@21656] shutdown OK
```

ProducerDemo supports sending general messages, delayed messages, and async messages.



3. Execute the `ConsumerDemo` file to receive messages.

Sending and receiving messages with topic model

1. Run the `PublishDemo` class to send messages with the topic model.
2. Run the `SubscriberDemo` class to receive messages with the topic model.

Sending and receiving transaction messages

1. Run the `ProducerTransactionDemo` class to send transaction messages.
2. Run the `SubscriberTransactionDemo` class to receive transaction messages.

">

NameServer Table

Region	Public Network Address	VPC Address
India	http://cmq-nameserver-in.tencentcloudapi.com	http://cmq-nameserver-vpc-in.api.tencentyun.com
Beijing	http://cmq-nameserver-bj.tencentcloudapi.com	http://cmq-nameserver-vpc-bj.api.tencentyun.com

Shanghai	http://cmq-nameserver-sh.tencentcloudapi.com	http://cmq-nameserver-vpc-sh.api.tencentyun.com
Guangzhou	http://cmq-nameserver-gz.tencentcloudapi.com	http://cmq-nameserver-vpc-gz.api.tencentyun.com
North America	http://cmq-nameserver-ca.tencentcloudapi.com	http://cmq-nameserver-vpc-ca.api.tencentyun.com
Chengdu	http://cmq-nameserver-cd.tencentcloudapi.com	http://cmq-nameserver-vpc-cd.api.tencentyun.com
Chongqing	http://cmq-nameserver-cq.tencentcloudapi.com	http://cmq-nameserver-vpc-cq.api.tencentyun.com
Hong Kong (China)	http://cmq-nameserver-hk.tencentcloudapi.com	http://cmq-nameserver-vpc-hk.api.tencentyun.com
Korea	http://cmq-nameserver-kr.tencentcloudapi.com	http://cmq-nameserver-vpc-kr.api.tencentyun.com
Russia	http://cmq-nameserver-ru.tencentcloudapi.com	http://cmq-nameserver-vpc-ru.api.tencentyun.com
Singapore	http://cmq-nameserver-sg.tencentcloudapi.com	http://cmq-nameserver-vpc-sg.api.tencentyun.com
Thailand	http://cmq-nameserver-th.tencentcloudapi.com	http://cmq-nameserver-vpc-th.api.tencentyun.com
East US	http://cmq-nameserver-use.tencentcloudapi.com	http://cmq-nameserver-vpc-use.api.tencentyun.com
West US	http://cmq-nameserver-usw.tencentcloudapi.com	http://cmq-nameserver-vpc-usw.api.tencentyun.com