ALIBABA CLOUD

# Alibaba Cloud

## Alibaba Cloud Message Queue

## Quick Start

Document Version: 20211130

Alibaba Cloud

# Legal disclaimer

Alibaba Cloud reminds you to carefully read and fully understand the terms and conditions of this legal disclaimer before you read or use this document. If you have read or used this document, it shall be deemed as your total acceptance of this legal disclaimer.

1. You shall download and obtain this document from the Alibaba Cloud website or other Alibaba Cloud-authorized channels, and use this document for your own legal business activities only. The content of this document is considered confidential information of Alibaba Cloud. You shall strictly abide by the confidentiality obligations. No part of this document shall be disclosed or provided to any third party for use without the prior written consent of Alibaba Cloud.

2. No part of this document shall be excerpted, translated, reproduced, transmitted, or disseminated by any organization, company or individual in any form or by any means without the prior written consent of Alibaba Cloud.

3. The content of this document may be changed because of product version upgrade, adjustment, or other reasons. Alibaba Cloud reserves the right to modify the content of this document without notice and an updated version of this document will be released through Alibaba Cloud-authorized channels from time to time. You should pay attention to the version changes of this document as they occur and download and obtain the most up-to-date version of this document from Alibaba Cloud-authorized channels.

4. This document serves only as a reference guide for your use of Alibaba Cloud products and services. Alibaba Cloud provides this document based on the "status quo", "being defective", and "existing functions" of its products and services. Alibaba Cloud makes every effort to provide relevant operational guidance based on existing technologies. However, Alibaba Cloud hereby makes a clear statement that it in no way guarantees the accuracy, integrity, applicability, and reliability of the content of this document, either explicitly or implicitly. Alibaba Cloud shall not take legal responsibility for any errors or lost profits incurred by any organization, company, or individual arising from download, use, or trust in this document. Alibaba Cloud shall not, under any circumstances, take responsibility for any indirect, consequential, punitive, contingent, special, or punitive damages, including lost profits arising from the use or trust in this document (even if Alibaba Cloud has been notified of the possibility of such a loss).

5. By law, all the contents in Alibaba Cloud documents, including but not limited to pictures, architecture design, page layout, and text description, are intellectual property of Alibaba Cloud and/or its affiliates. This intellectual property includes, but is not limited to, trademark rights, patent rights, copyrights, and trade secrets. No part of this document shall be used, modified, reproduced, publicly transmitted, changed, disseminated, distributed, or published without the prior written consent of Alibaba Cloud and/or its affiliates. The names owned by Alibaba Cloud shall not be used, published, or reproduced for marketing, advertising, promotion, or other purposes without the prior written consent of Alibaba Cloud. The names owned by Alibaba Cloud include, but are not limited to, "Alibaba Cloud", "Aliyun", "HiChina", and other brands of Alibaba Cloud and/or its affiliates, which appear separately or in combination, as well as the auxiliary signs and patterns of the preceding brands, or anything similar to the company names, trade names, trademarks, product or service names, domain names, patterns, logos, marks, signs, or special descriptions that third parties identify as Alibaba Cloud and/or its affiliates.

6. Please directly contact Alibaba Cloud for any errors of this document.

# Document conventions

| Style | Description | Example |
| --- | --- | --- |
| Danger | A danger notice indicates a situation that will cause major system changes, faults, physical injuries, and other adverse results. | **Danger:** Resetting will result in the loss of user configuration data. |
| Warning | A warning notice indicates a situation that may cause major system changes, faults, physical injuries, and other adverse results. | **Warning:** Restarting will cause business interruption. About 10 minutes are required to restart an instance. |
| Notice | A caution notice indicates warning information, supplementary instructions, and other content that the user must understand. | **Notice:** If the weight is set to 0, the server no longer receives new requests. |
| Note | A note indicates supplemental instructions, best practices, tips, and other content. | **Note:** You can use Ctrl + A to select all files. |
| > | Closing angle brackets are used to indicate a multi-level menu cascade. | Click **Settings**> **Network**> **Set network type**. |
| **Bold** | Bold formatting is used for buttons , menus, page names, and other UI elements. | Click **OK**. |
| `Courier font` | Courier font is used for commands | Run the `cd /d C:/window` command to enter the Windows system folder. |
| *Italic* | Italic formatting is used for parameters and variables. | `bae log list --instanceid` *Instance_ID* |
| [] or [a\|b] | This format is used for an optional value, where only one item can be selected. | `ipconfig [-all\|-t]` |
| {} or {a\|b} | This format is used for a required value, where only one item can be selected. | `switch {active\|stand}` |

# Table of Contents

# 1.Overview

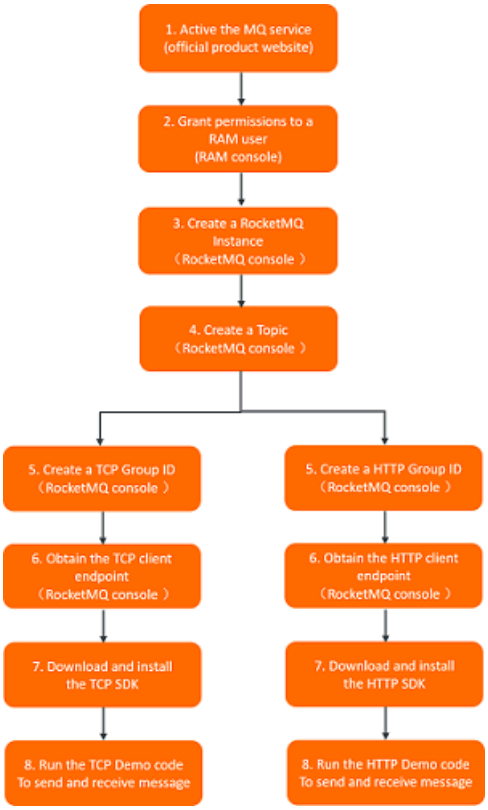
Message Queue for Apache RocketMQ
provides the TCP client SDK and HTTP client SDK for multiple programming languages. You can use the SDKs to send and subscribe to different types of messages. This topic describes the procedure and notes for using the TCP client SDK and HTTP client SDK for multiple programming languages to send and subscribe to normal messages.

## Background information

Message Queue for Apache RocketMQ
provides four types of messages. For more information, see Overview of features. In this topic, normal messages are used as an example. The topic that you create by following the procedure in this topic cannot be used to send or subscribe to other types of messages. Each topic can be used to send and subscribe to messages of a specific type.

## Procedure

The following figure shows the procedure based on the protocol that you select.




## Usage notes

- Message Queue for Apache RocketMQ
  provides different SDKs for different protocols that include TCP and HTTP used to send and subscribe to messages. You cannot specify the same group ID in the code of the TCP client SDK and the code of the HTTP client SDK at the same time.
- A
  Message Queue for Apache RocketMQ
  instance has a TCP endpoint and an HTTP endpoint. If you send and subscribe to messages over TCP, you must use the TCP client SDK and the TCP endpoint. If you send and subscribe to messages over

HTTP, you must use the HTTP client SDK and the HTTP endpoint.

- Public endpoints are available for TCP clients only in the **Internet** region. In other regions, only internal endpoints are available. Public and internal endpoints are available for HTTP clients in all regions.
- If your application uses
  Message Queue for Apache RocketMQ
  across regions, we recommend that you use HTTP.
- More features will be developed for the HTTP client SDK to align with those of the TCP client SDK. For more information about how to use the HTTP client SDK, see Release notes.

## What to do next

- Use the TCP client SDK to send and subscribe to messages
- Use the HTTP client SDK to send and subscribe to messages

# 2.Call TCP SDKs to send and subscribe to messages

## 2.1. Activate Message Queue for Apache RocketMQ and grant permissions

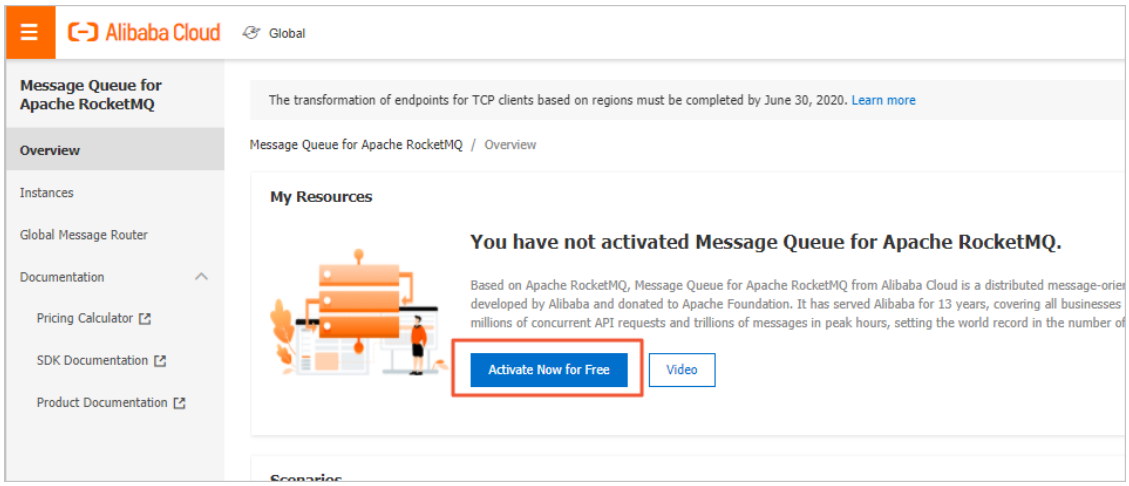
Before you use
Message Queue for Apache RocketMQ
, you must activate it on the Alibaba Cloud official website. If your account is a Resource Access Management (RAM) user, you must grant permissions to the RAM user before you can use the console or API to access the corresponding
Message Queue for Apache RocketMQ
resources and use the resources to send and subscribe to messages by using SDKs.

### Prerequisites

An Alibaba Cloud account is created and real-name verification is complete. For more information, see Sign up with Alibaba Cloud.

### Step 1: Activate Message Queue for Apache RocketMQ

1. Open the
   Message Queue for Apache RocketMQ
   product details page.
2. Click **Log In** in the upper-right corner of the page.
3. On the **Sign In** page, enter your Alibaba Cloud account and password, and click **Sign In**.
4. On the product details page, click **Console**.
   You are redirected to the
   Message Queue for Apache RocketMQ
   console.
5. On the **Overview** page, click **Activate Now for Free**.




6. On the service activation page, read the content of the order and the service agreement, select **Message Queue (MQ) Terms of Service**, and then click **Activate Now**.

## Step 2: (Required) Grant permissions to a RAM user

1. Log on to the RAM console.
2. In the left-side navigation pane, choose **Identities > Users**.
3. On the **Users** page, find the RAM user to which you want to grant permissions and click **Add Permissions** in the **Actions** column.
4. In the **Add Permissions** panel, click the policy that you want to grant to the RAM user. Then, click **OK**.

   Message Queue for Apache RocketMQ provides the following system policies. You can grant related permissions to the RAM user based on the permission scope.

| Policy | Description |
| --- | --- |
| AliyunMQFullAccess | The permissions to manage Message Queue for Apache RocketMQ. Such permissions are equivalent to the permissions that an Alibaba Cloud account has. A RAM user to which this policy is attached can send and subscribe to all messages and use all the features of the console. |
| AliyunMQPubOnlyAccess | The message sending permissions of Message Queue for Apache RocketMQ. A RAM user to which this policy is attached can use all the resources of the relevant Alibaba Cloud account to send messages by using SDKs. |
| AliyunMQSubOnlyAccess | The message subscription permissions of Message Queue for Apache RocketMQ. A RAM user to which this policy is attached can use all the resources of the relevant Alibaba Cloud account to subscribe to messages by using SDKs. |
| AliyunMQReadOnlyAccess | The read-only permissions on Message Queue for Apache RocketMQ. A RAM user to which this policy is attached can only read resource information by using the console or by calling API operations. |

> **Note** System permissions cover a large permission scope. For example, after a RAM user is granted `AliyunMQFullAccess` that represents full permissions, the RAM user can manage all resources of Message Queue for Apache RocketMQ.
>
> Message Queue for Apache RocketMQ provides more custom policies for fine-grained authorization on a specific type of resource. For example, you can grant RAM users only permissions to manage topics in the console. For more information about custom policies, see Policies.

5. In the **Add Permissions** panel, click **Complete**.

### What's next

You can click **Console** to create resources. For more information, see Create resources.

# 2.2. Create resources

Before you call a TCP client SDK to send and subscribe to messages, you must create required resources in the
Message Queue for Apache RocketMQ
console. You must specify the resource parameters when you call the SDK.

### Prerequisites

Activate Message Queue for Apache RocketMQ and grant permissions

### Context

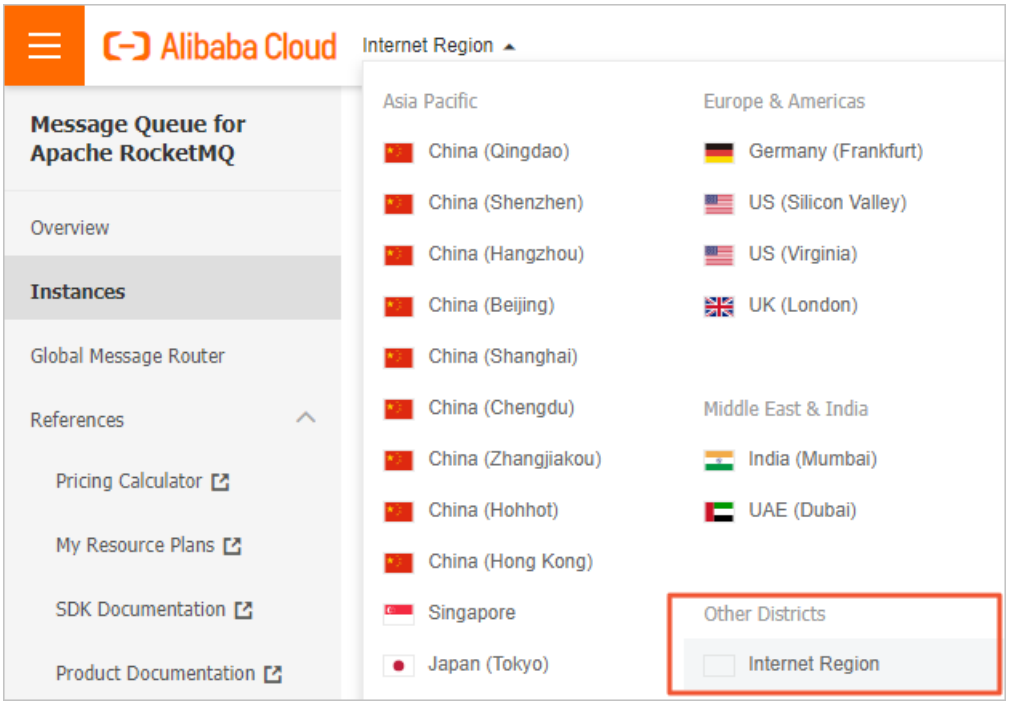
Message Queue for Apache RocketMQ
provides SDKs for multiple programming languages to send and subscribe to messages over TCP and HTTP. To use
Message Queue for Apache RocketMQ
, you must create an instance, a topic, and a group ID. Due to client differences, a group ID cannot be used for both TCP clients and HTTP clients. Therefore, you must create group IDs for TCP clients and HTTP clients separately.

### Network access limits

When you use

Message Queue for Apache RocketMQ
, take note of the following network access limits:

- A topic can be accessed only by a producer or consumer whose group ID is created on the same instance in the same region as the topic. For example, if a topic is created on Instance A in the **China (Hangzhou)** region, the topic can be accessed only by a producer or consumer whose group ID is created on Instance A in the **China (Hangzhou)** region.
- If you use an on-premises server and need to access
  Message Queue for Apache RocketMQ
  over the Internet, you must create a topic and group ID on an instance in the **Internet** region.




Producer and consumer clients can be deployed on an on-premises server or an Elastic Compute Service (ECS) instance in a region. You must make sure that the on-premises server or ECS instance can access the Internet.

For more information about regions, see Regions and zones.

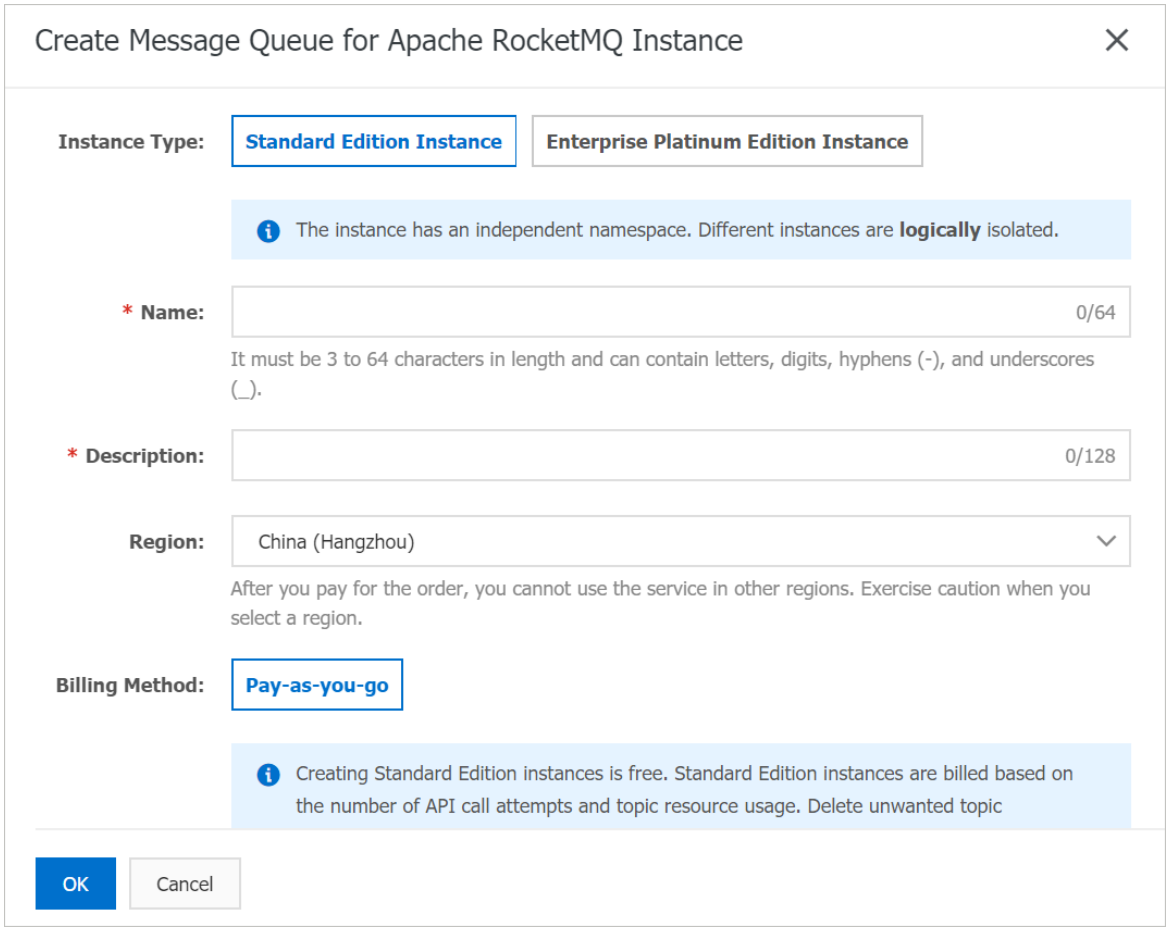
## Create an instance

An instance is a virtual machine (VM) resource of

Message Queue for Apache RocketMQ

. An instance stores topics and group IDs.

1. Log on to the Message Queue for Apache RocketMQ console.
2. In the left-side navigation pane, click **Instances**.
3. In the top navigation bar, select a region, such as **China (Hangzhou)**.
4. On the **Instances** page, click **Create Instance**.
5. In the **Create Message Queue for Apache RocketMQ Instance** panel, set **Instance Type**, **Instance Name**, and **Description**, and then click **OK**.




For more information about the billing of Message Queue for Apache RocketMQ Standard Edition and Enterprise Platinum Edition instances, see Overview.

After the instance is created, it is displayed on the Instances page.

## Create a topic

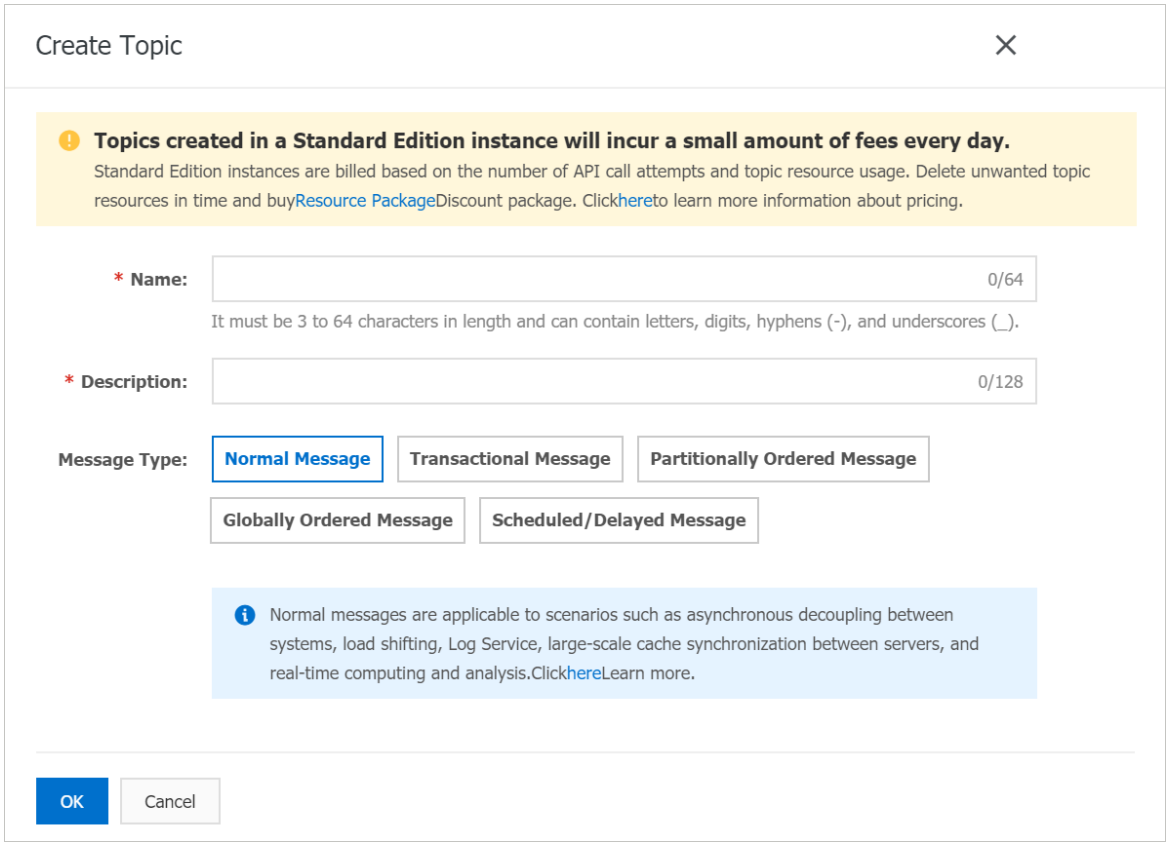
A topic is a first-level identifier that is used to classify messages in

Message Queue for Apache RocketMQ

. For example, you can create a topic named Topic_Trade for transactional messages. Message producers send messages to Topic_Trade and message consumers subscribe to Topic_Trade to consume the messages. Topics must meet the following requirements:

- A topic cannot be used across instances. For example, Topic A created in Instance A is unavailable in Instance B.
- Each topic name must be unique in an instance.
- We recommend that you create different topics to send different types of messages. For example, you can create Topic A for normal messages, Topic B for transactional messages, and Topic C for scheduled and delayed messages.

1. On the page of the instance, click **Topics** in the left-side navigation pane.
2. On the **Topics** page, click **Create Topic**.
3. In the **Create Topic** panel, set **Name** and **Description**, set **Message Type** to **Normal Message**, and then click **OK**.




> **Note** This topic describes how to send and subscribe to normal messages. Therefore, the topic that you create for normal messages by using the preceding procedure cannot be used to send or subscribe to other types of messages. Each topic can be used to send and subscribe to messages of a specified type. For more information, see Overview .

The created topic appears in the topic list.

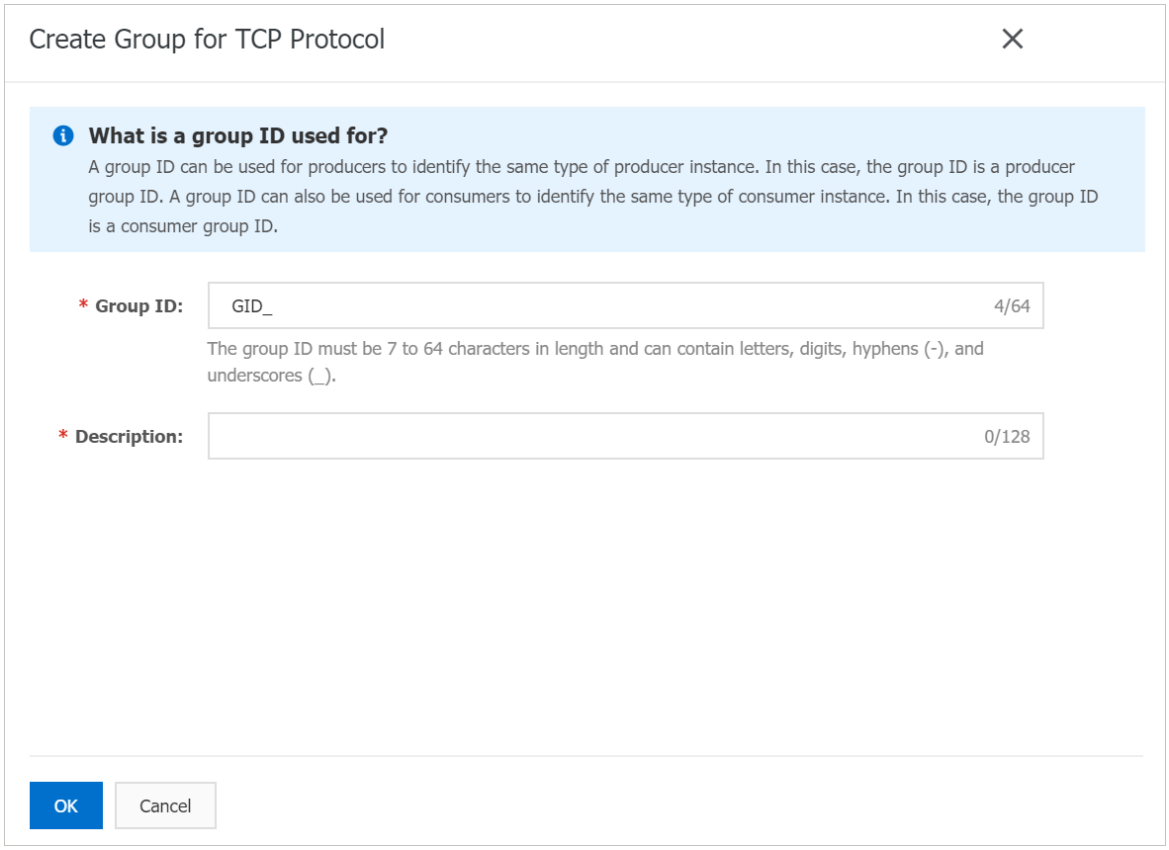
## Create a group ID

After you create an instance and a topic, you must create a group ID for the message consumer or producer. Group IDs must meet the following requirements:

- Each group ID must be unique in an instance.
- Group IDs have an N:N relationship with topics. A consumer can subscribe to multiple topics and a topic can be subscribed to by multiple consumers. A producer can send messages to multiple topics and a topic can receive messages from multiple producers.

> **Note** A group ID is required for consumers but is optional for producers.

1. On the page of the instance, click **Groups** in the left-side navigation pane.
2. On the **Groups** page, choose **TCP > Create Group ID**.
3. In the **Create Group for TCP Protocol** panel, set **Group ID** and **Description**, and then click **OK**.

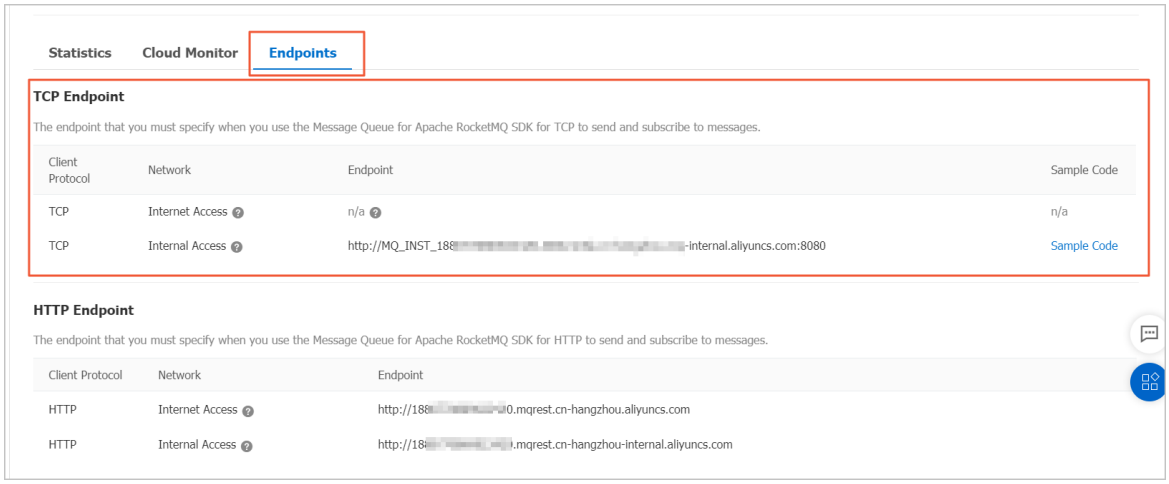



The created group ID appears in the group ID list.

## Obtain an endpoint

After you create an instance in the console, you must obtain an endpoint of the instance in the Message Queue for Apache RocketMQ console. To send and subscribe to messages, you must configure the endpoint for the producer and consumer. Then, the producer and consumer can access services in the corresponding instance or region.

1. On the page of the instance, click **Instance Details** in the left-side navigation pane.
2. On the **Instance Details** page, click the **Endpoints** tab.
3. In the **TCP Endpoint** section, move the pointer over the endpoint and click the endpoint to copy it.




Usage notes of endpoints:

- The TCP endpoint displayed in the console is the endpoint of a specific instance in a region. Different instances in the same region have different endpoints.
- Only the instances in the Internet region have public TCP endpoints. Instances in other regions have only internal TCP endpoints.
- TCP endpoints cannot be used across regions. For more information, see Announcement on the settings of internal TCP endpoints.

> **Note** The HTTP endpoint and TCP endpoint of the same instance cannot be interchangeably used.

After you complete the preceding operations, you can run the sample code and use

Message Queue for Apache RocketMQ
to send and subscribe to messages.

## What's next

Call TCP client SDKs to send and subscribe to normal messages

# 2.3. Call TCP SDKs to send and subscribe to normal messages

After you create all the resources in the console, you can call
Message Queue for Apache RocketMQ
TCP SDK to send and subscribe to normal messages.

## Prerequisites

- Create resources

> **Note** In this topic, normal messages are used in the example. The topic that is created for normal messages cannot be used to send or subscribe to other types of messages, such as scheduled messages, delayed messages, ordered messages, and transactional messages. You must create topics based on message types.

- Obtain an AccessKey pair

## Download and install a TCP SDK

> **Note** Commercial SDKs provide more features and higher stability than open source SDKs. We recommend that you use commercial SDKs provided by Message Queue for Apache RocketMQ. Open source SDKs are used only when you migrate open source Apache RocketMQ to Message Queue for Apache RocketMQ but do not want to modify the code.

Message Queue for Apache RocketMQ provides the following commercial TCP SDKs. Obtain the client SDK for a specific programming language as needed.

## Call TCP SDKs to send normal messages

After you obtain the client SDK for a specific programming language, you can run the following sample code to send normal messages:

You can also start your instance by performing the following steps: Log on to the Message Queue for Apache RocketMQ console. Find the created instance and click **More** in the **Actions** column. Select **Quick Start** from the drop-down list.

Java | .NET | C/C++

## Check whether messages are sent

After a message is sent, you can check its status in the Message Queue for Apache RocketMQ console by performing the following operations:

1. On the Instance Details page, click **Message Query** in the left-side navigation pane.
2. On the **Message Query** page, select a query method and specify the parameters as required, and then click **Search**.
   **Stored At** indicates the time when the Message Queue for Apache RocketMQ broker stores the message. If a message can be queried, the message has been sent to the Message Queue for Apache RocketMQ broker.

> **Notice** This step demonstrates the scenario where Message Queue for Apache RocketMQ is used for the first time and the consumer has not been started. Therefore, no consumption data is displayed in the console. To start the consumer and subscribe to messages, see the next section. For more information about the message status, see Query messages and Query a message trace.

## Call TCP SDKs to subscribe to normal messages

After a normal message is sent, you must start a consumer to subscribe to the message. You can use the following sample code for a specific programming language based on your business requirements to start the consumer and test the message subscription feature. Set the parameters based on the instructions.

Java | .NET | C/C++

## Check whether the message subscription is successful

1. On the Instance Details page of the instance, click **Groups** in the left-side navigation pane.
2. On the **Groups** page, click the **TCP** tab.
3. Find the Group ID whose subscription status that you want to view, and click **Details** in the **Actions** column.
   If the value of **Consumer Status** is **Online** and the value of the Is Subscription Consistent parameter is Yes, the message subscription is successful.

# 3.Call HTTP SDKs to send and subscribe to messages

## 3.1. Activate Message Queue for Apache RocketMQ and grant permissions

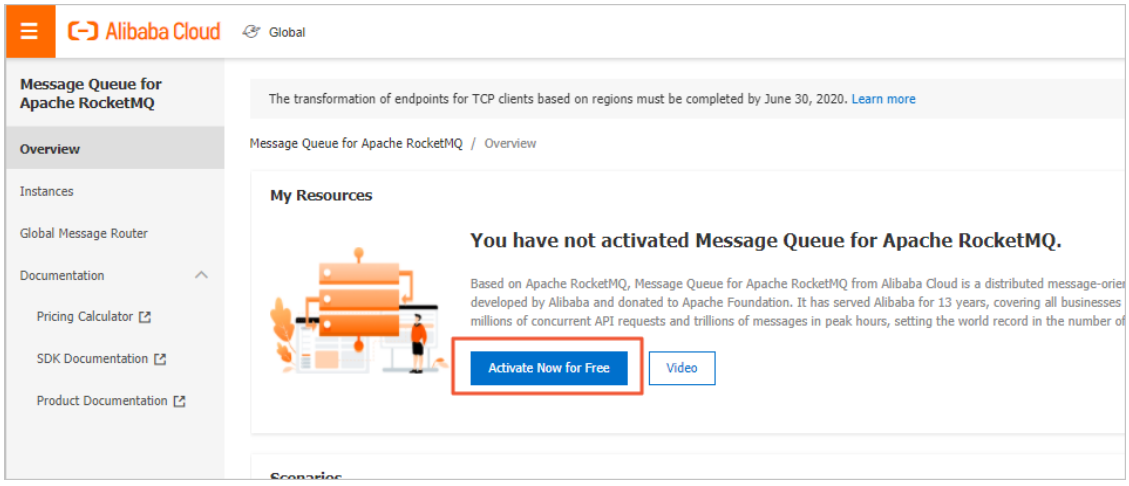
Before you use
Message Queue for Apache RocketMQ
, you must activate it on the Alibaba Cloud official website. If your account is a Resource Access Management (RAM) user, you must grant permissions to the RAM user before you can use the console or API to access the corresponding
Message Queue for Apache RocketMQ
resources and use the resources to send and subscribe to messages by using SDKs.

### Prerequisites

An Alibaba Cloud account is created and real-name verification is complete.

### Step 1: Activate Message Queue for Apache RocketMQ

1. Open the
   Message Queue for Apache RocketMQ
   product details page.
2. Click **Log In** in the upper-right corner of the page.
3. On the **Sign In** page, enter your Alibaba Cloud account and password, and click **Sign In**.
4. On the product details page, click **Console**.
   You are redirected to the
   Message Queue for Apache RocketMQ
   console.
5. On the **Overview** page, click **Activate Now for Free**.




6. On the service activation page, read the content of the order and the service agreement, select **Message Queue (MQ) Terms of Service**, and then click **Activate Now**.

## Step 2: (Required) Grant permissions to a RAM user

1. Log on to the RAM console.
2. In the left-side navigation pane, choose **Identities > Users**.
3. On the **Users** page, find the RAM user to which you want to grant permissions and click **Add Permissions** in the **Actions** column.
4. In the **Add Permissions** panel, click the policy that you want to grant to the RAM user. Then, click **OK**.

   Message Queue for Apache RocketMQ provides the following system policies. You can grant related permissions to the RAM user based on the permission scope.

| Policy | Description |
| --- | --- |
| AliyunMQFullAccess | The permissions to manage Message Queue for Apache RocketMQ. Such permissions are equivalent to the permissions that an Alibaba Cloud account has. A RAM user to which this policy is attached can send and subscribe to all messages and use all the features of the console. |
| AliyunMQPubOnlyAccess | The message sending permissions of Message Queue for Apache RocketMQ. A RAM user to which this policy is attached can use all the resources of the relevant Alibaba Cloud account to send messages by using SDKs. |
| AliyunMQSubOnlyAccess | The message subscription permissions of Message Queue for Apache RocketMQ. A RAM user to which this policy is attached can use all the resources of the relevant Alibaba Cloud account to subscribe to messages by using SDKs. |
| AliyunMQReadOnlyAccess | The read-only permissions on Message Queue for Apache RocketMQ. A RAM user to which this policy is attached can only read resource information by using the console or by calling API operations. |

> **Note** System permissions cover a large permission scope. For example, after a RAM user is granted `AliyunMQFullAccess` that represents full permissions, the RAM user can manage all resources of Message Queue for Apache RocketMQ.
>
> Message Queue for Apache RocketMQ provides more custom policies for fine-grained authorization on a specific type of resource. For example, you can grant RAM users only permissions to manage topics in the console. For more information about custom policies, see Policies.

5. In the **Add Permissions** panel, click **Complete**.

## What's next

You can click **Console** to create resources. For more information, see Create resources.

# 3.2. Create resources

Before you call an HTTP SDK to send and subscribe to messages, you must create required resources in the
Message Queue for Apache RocketMQ
console.

## Prerequisites

Activate Message Queue for Apache RocketMQ and grant permissions

## Context

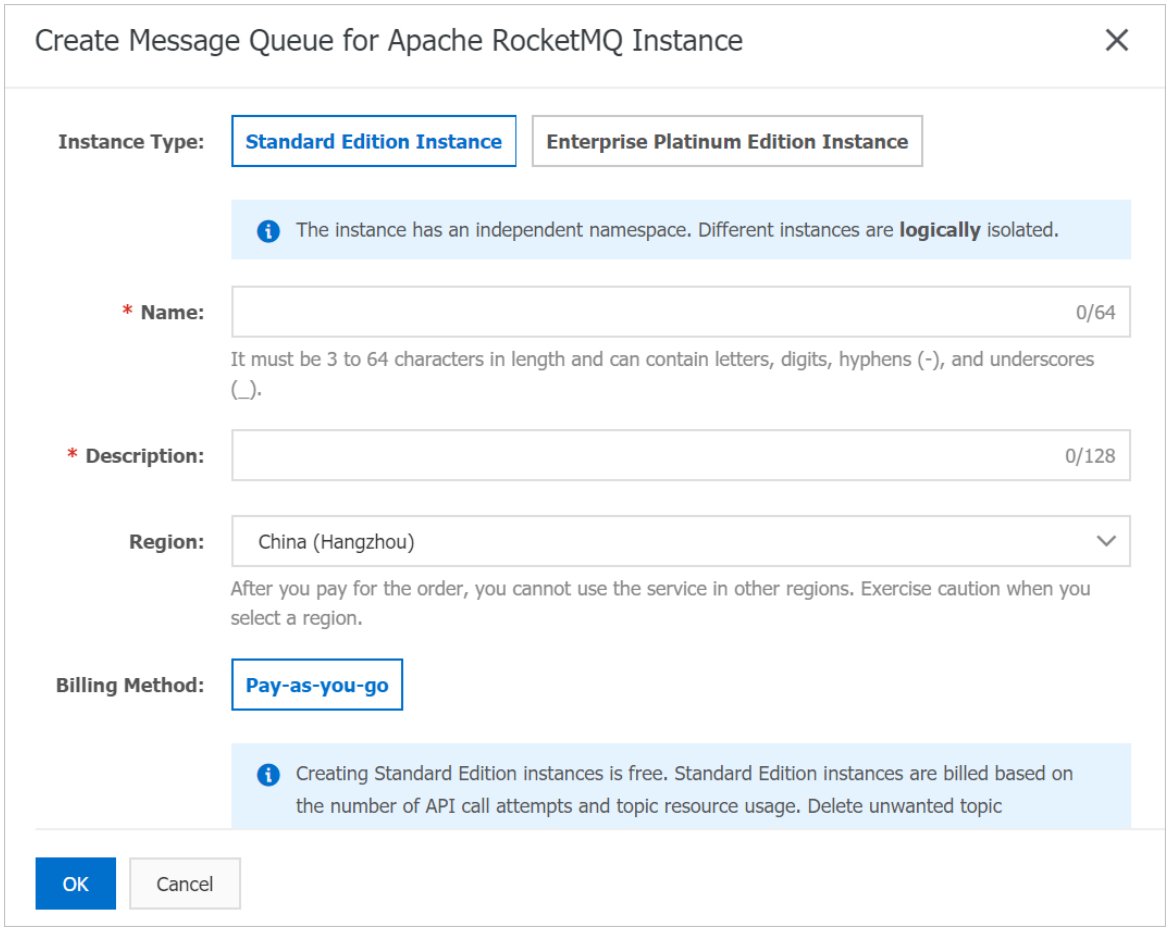
Message Queue for Apache RocketMQ
provides SDKs for multiple programming languages to send and subscribe to messages over TCP and HTTP. To use
Message Queue for Apache RocketMQ
, you must create an instance, a topic, and a group ID. Due to client differences, a group ID cannot be used for both TCP clients and HTTP clients. Therefore, you must create group IDs for TCP clients and HTTP clients separately.

## Create an instance

An instance is a virtual machine (VM) resource of

Message Queue for Apache RocketMQ
. An instance stores the topics and group IDs.

1. Log on to the
   Message Queue for Apache RocketMQ
   console.
2. In the left-side navigation pane, click **Instances**.
3. In the top navigation bar, select a region, for example, **China (Hangzhou)**.
4. On the **Instances** page, click **Create Instance**.
5. In the **Create Message Queue for Apache RocketMQ Instance** dialog box, select an **Instance Type**, enter an **Instance Name** and **Description**, and then click **OK**.




For information about the billing of Message Queue for Apache RocketMQ Standard Edition and Platinum Edition instances, see Overview.

After the instance is created, it is displayed on the Instances page.

## Create a topic

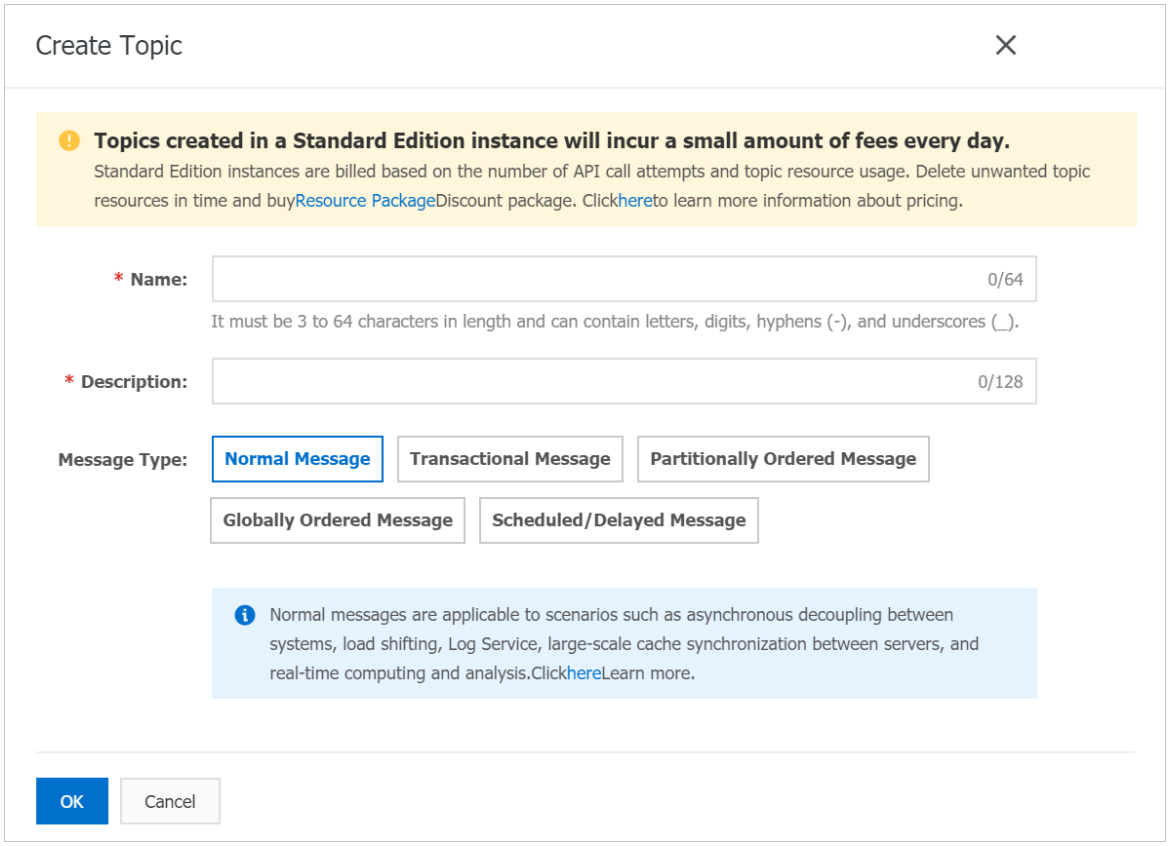
A topic is a first-level identifier that is used to classify messages in

Message Queue for Apache RocketMQ

. For example, you can create a topic named Topic_Trade for transactional messages. The message producers send messages to Topic_Trade and the message consumers subscribe to Topic_Trade to consume the messages.

- A topic cannot be used across instances. For example, Topic A created in Instance A is unavailable in Instance B.
- Each topic name must be unique in an instance.
- We recommend that you create different topics to send different types of messages. For example, you can create Topic A for common messages, Topic B for transactional messages, and Topic C for scheduled and delayed messages.

1. On the instance details page, click **Topics** in the left-side navigation pane.
2. On the **Topics** page, click **Create Topic**.
3. In the **Create Topic** dialog box, enter a name and **Description**, set the **Message Type** to **Common Message**, and then click **OK**.




> **Note** This topic uses common messages as an example. You can use the preceding procedure to create a topic that processes common messages. The topic you created cannot be used to send or receive other types of messages. Each topic can send and receive messages of a specified type. For more information, see Message types.

The created topic is displayed in the topic list.

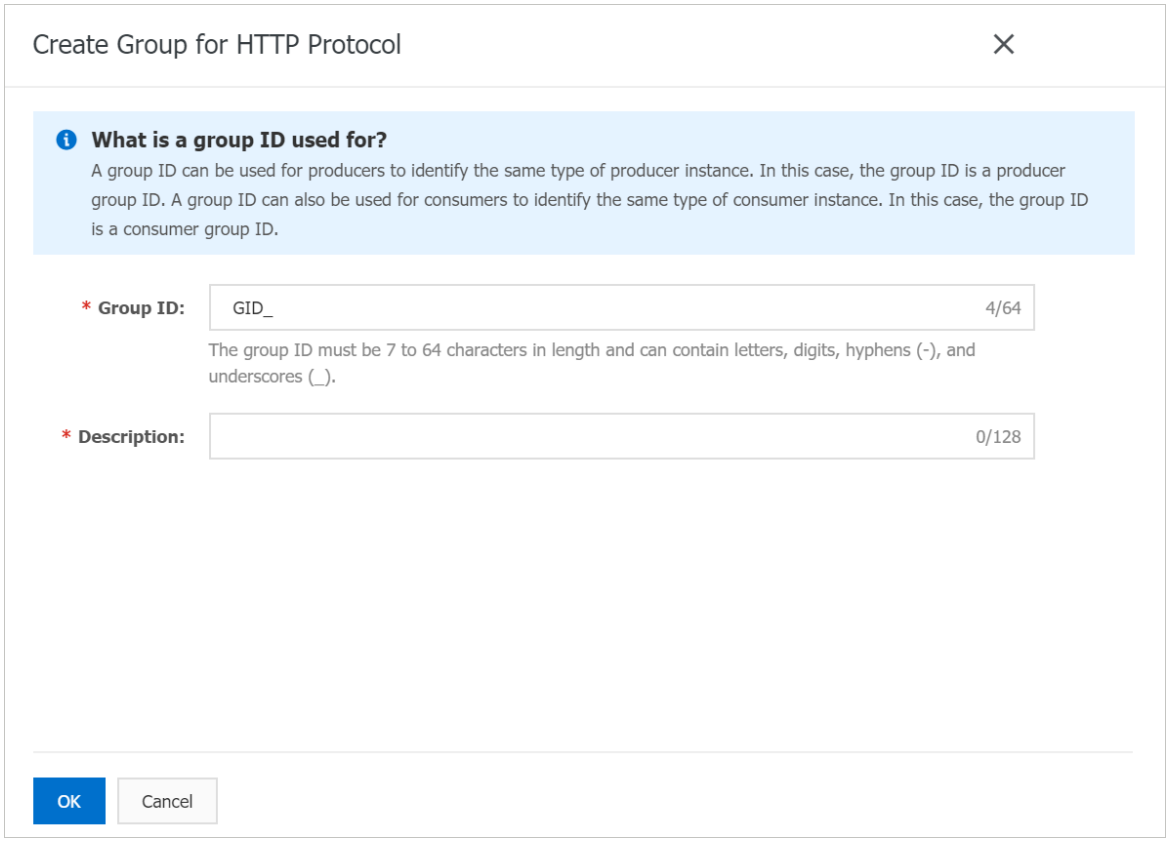
## Create a group ID

After you have created an instance and a topic, you need to create a group ID for the message consumer or producer.

- A group ID must be unique within an instance.
- Group IDs and topics implement N:N mapping. A consumer can subscribe to multiple topics and a topic can be subscribed to by multiple consumers. A producer can send messages to multiple topics and a topic can receive messages from multiple producers.

> **Note** A group ID is required for consumers but is optional for producers.

1. On the Instance Details page, click **Groups** in the left-side navigation pane.
2. On the **Groups** page, choose **HTTP > Create Group ID**.
3. In the **Create Group ID** dialog box, enter a **Group ID** and **Description**, and then click **OK**.

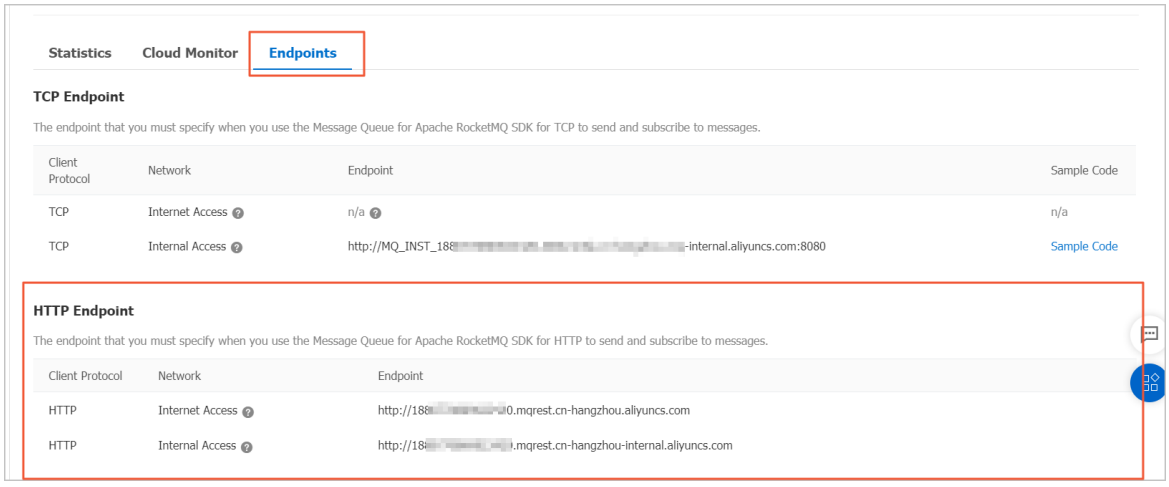



The created group ID is displayed in the group ID list.

## Obtain an endpoint

After you create an instance in the console, you must obtain an endpoint of the instance in the Message Queue for Apache RocketMQ console. To send and subscribe to messages, you must configure the endpoint for the producer and consumer. Then, the producer and consumer can access services in the corresponding instance or region.

1. On the page of the instance, click **Instance Details** in the left-side navigation pane.
2. On the **Instance Details** page, click the **Endpoints** tab.
3. In the **HTTP Endpoint** section, move the pointer over the endpoint and click the endpoint to copy it.



The HTTP endpoint displayed in the console is the endpoint of a region, instead of a specific instance. You must configure an ID for the instance when you send and subscribe to messages.

> **Note** The HTTP endpoint and TCP endpoint of the same instance cannot be interchangeably used.

After you complete the preceding operations, you can run the sample code and use

Message Queue for Apache RocketMQ
to send and subscribe to messages.

## What's next

Call HTTP SDKs to send and subscribe to normal messages

# 3.3. Call HTTP SDKs to send and receive messages

After you create all the resources in the console, you can call an HTTP SDK to send and receive messages in
Message Queue for Apache RocketMQ
.

## Prerequisites

- Create resources

> **Note** In this topic, normal messages are used in the example. The topic that is created for normal messages cannot be used to send or subscribe to other types of messages, such as scheduled messages, delayed messages, ordered messages, and transactional messages. You must create topics based on message types.

- Create an AccessKey pair

## Download and install an HTTP SDK

Message Queue for Apache RocketMQ
provides the following HTTP SDKs for multiple programming languages. Download and install the client SDK for a specific language as needed.

**Java SDK**

- Access description
- Release notes

**PHP SDK**

- Access description
- Release notes

Go SDK
- Access description
- Release notes

Python SDK
- Access description
- Release notes

Node.js SDK
- Access description
- Release notes

C# SDK
- Access description
- Release notes

C++ SDK
- Access description
- Release notes

## Call HTTP SDKs to send messages

After you obtain the client SDK for a specific language, you can run the following sample code to send messages:

You can also start your instance by performing the following steps: Log on to the Message Queue for Apache RocketMQ console. Find the created instance and click **More** in the **Actions** column. Select **Quick Start** from the drop-down list.

Java | Go | PHP | Python | Node.js | C++ | C#

## Call HTTP SDKs to receive messages

After a message is sent, you must enable a consumer client to receive the message. You can use the following sample code for a specific programming language based on your business requirements. Set the parameters based on the instructions.

Java | Go | PHP | Python | Node.js | C++ | C#

```
import com.aliyun.mq.http.MQClient;
import com.aliyun.mq.http.MQConsumer;
import com.aliyun.mq.http.common.AckMessageException;
```

```
import com.aliyun.mq.http.model.Message;

import java.util.ArrayList;
import java.util.List;

public class Consumer {

  public static void main(String[] args) {
    MQClient mqClient = new MQClient(
        // Set an HTTP endpoint.
        "${HTTP_ENDPOINT}",
        // The AccessKey ID that you created in the Alibaba Cloud Management Console.
        "${ACCESS_KEY}",
        // The AccessKey secret that you created in the Alibaba Cloud Management Console.
        "${SECRET_KEY}"
    );

    // The topic of the message.
    final String topic = "${TOPIC}";
    // The group ID (consumer ID) that you created in the console.
    final String groupId = "${GROUP_ID}";
    // The instance ID of the topic. The default value is NULL.
    final String instanceId = "${INSTANCE_ID}";

    final MQConsumer consumer;
    if (instanceId != null && instanceId != "") {
      consumer = mqClient.getConsumer(instanceId, topic, groupId, null);
    } else {
      consumer = mqClient.getConsumer(topic, groupId);
    }

    // Cyclically consume messages in the current thread. We recommend that you use multiple
threads to concurrently consume messages.
    do {
      List<Message> messages = null;

      try {
        // Consume messages in long-polling mode.
        // In long-polling mode, if no message on the topic is available for consumption, the request
is hung on the server for 3 seconds. If any message is available for consumption within the duration, a
response is immediately sent to the client.
        messages = consumer.consumeMessage(
            3,// A maximum of 3 messages can be consumed at a time. You can set the value up to
16.
            3 // The duration of a long-polling cycle is 3 seconds. You can set the value up to 30
seconds.
        );
      } catch (Throwable e) {
        e.printStackTrace();
        try {
          Thread.sleep(2000);
        } catch (InterruptedException e1) {
          e1.printStackTrace();
        }
      }
      // No messages.
      if (messages == null || messages.isEmpty()) {
        System.out.println(Thread.currentThread().getName() + ": no new message, continue! ");
        continue;
      }

      // Specify business logic.
      for (Message message : messages) {
        System.out.println("Receive message: " + message);
```

```
            System.out.println("Receive message: " + message);
        }

        // If the consumption of a message is not confirmed before the time that is specified by the
Message.nextConsumeTime parameter, the message will be consumed again.
        // A unique timestamp is specified for the handle of a message each time the message is
consumed.
        {
          List<String> handles = new ArrayList<String>();
          for (Message message : messages) {
            handles.add(message.getReceiptHandle());
          }

          try {
            consumer.ackMessage(handles);
          } catch (Throwable e) {
            // The confirmation of message consumption may fail due to the timeout of message
handles.
            if (e instanceof AckMessageException) {
              AckMessageException errors = (AckMessageException) e;
              System.out.println("Ack message fail, requestId is:" + errors.getRequestId() + ", fail
handles:");
              if (errors.getErrorMessages() != null) {
                for (String errorHandle : errors.getErrorMessages().keySet()) {
                  System.out.println("Handle:" + errorHandle + ", ErrorCode:" +
errors.getErrorMessages().get(errorHandle).getErrorCode()
                      + ", ErrorMsg:" + errors.getErrorMessages().get(errorHandle).getErrorMessage());
                }
              }
              continue;
            }
            e.printStackTrace();
          }
        }
      } while (true);
    }
}
```

## What's next

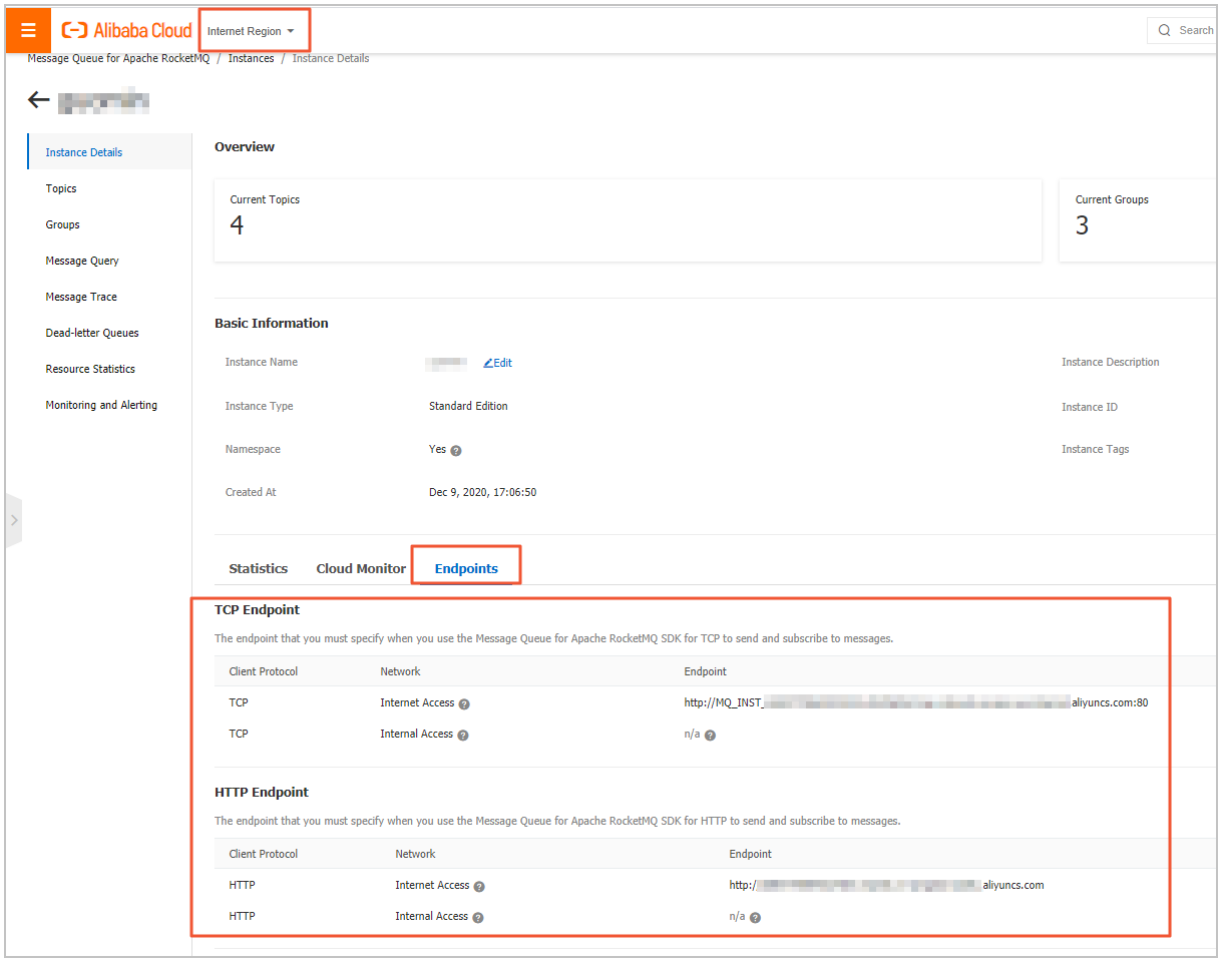
You can query messages and their traces to verify whether messages are consumed. For more information, see Query messages and Query a message trace.

# 4.Quick start FAQ

## 4.1. Can I access over the Internet?

Yes, you can access
Message Queue for Apache RocketMQ
over the Internet.

Message Queue for Apache RocketMQ
provides a region named Internet for your access over the Internet. When you create an instance, a topic, and a group ID in the
Message Queue for Apache RocketMQ
console, select **Internet Region**. Then, your client can access the topic by using the public TCP or HTTP endpoint of
Message Queue for Apache RocketMQ
.




The following table describes the endpoints of Message Queue for Apache RocketMQ.

| Region | TCP endpoint | HTTP endpoint |
| --- | --- | --- |
| Internet | Only the public endpoint is available. | Only the public endpoint is available. |

| Region | TCP endpoint | HTTP endpoint |
| --- | --- | --- |
| Regions other than the Internet region | Only the internal endpoint is available. | Both the public and internal endpoints are available. |

However, the Internet environment is complex and less stable than the internal network. In the production environment, you can deploy your application on an Elastic Compute Service (ECS) instance in a region in the internal network to send and receive messages. Regions in the internal network provide higher availability.

For more information about how to send and receive messages by using

Message Queue for Apache RocketMQ
, see the following two topics:

- Use TCP SDK for Java to send and receive messages
- Use HTTP SDK for Java to send and receive messages

# 4.2. Is Message Queue for Apache RocketMQ always free of duplicate messages?

Messages are not duplicated in most cases. As a distributed messaging middleware, Message Queue for Apache RocketMQ cannot ensure that messages are not duplicated when exceptions (such as network jitter or application processing timeout) occur. However, Message Queue for Apache RocketMQ can ensure that no messages are lost.

# 4.3. How long can messages be retained on the broker?

All types of published messages are retained on the

Message Queue for Apache RocketMQ

broker for up to three days. The messages are deleted after three days no matter whether they are consumed. We recommend that you use the monitoring and alerting feature to monitor the message consumption status in real time, and manually intervene upon alerts. For more information, see Overview.

# 4.4. What is the maximum length of the message body in Message Queue for Apache RocketMQ?

The maximum length of the message body in Message Queue for Apache RocketMQ varies with the message type. The following shows the details:

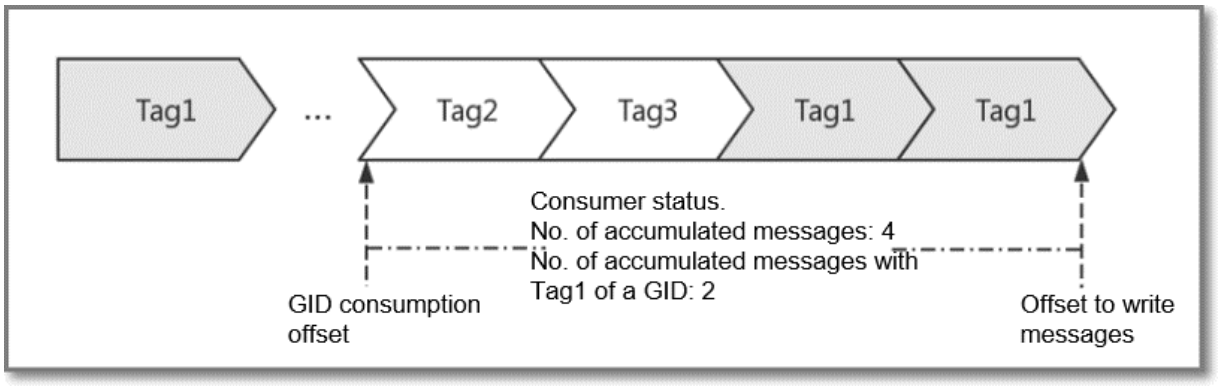
- A normal or ordered message: 4 MB
- A transactional, scheduled, or delayed message: 64 KB

# 4.5. Does the value of Accumulation Amount in the console include the number of all tagged messages under topics?

Yes.

The message producer sends messages of all tags to the same topic. Messages are arranged in the queue in order and a message write point is maintained. When a group ID is started, it specifies the tags to be subscribed to and obtains the current consumption offset from the broker. The broker traverses messages in the queue from the consumption offset of the current group ID. If the tag of a message is consistent with the tag subscribed by the group ID, the broker delivers the message to the group ID. Otherwise, the broker skips this message.

As shown in the following figure, the consumption offset of the group ID moves forward. Messages with Tag2 and Tag3 are filtered out by the broker. The message with Tag1 is required by the group ID and will be delivered to the group ID.




Therefore, the accumulation amount you saw after choosing **Consumer Status > Accumulation Amount** in the console is the unfiltered accumulation amount, which contains messages with all tags.