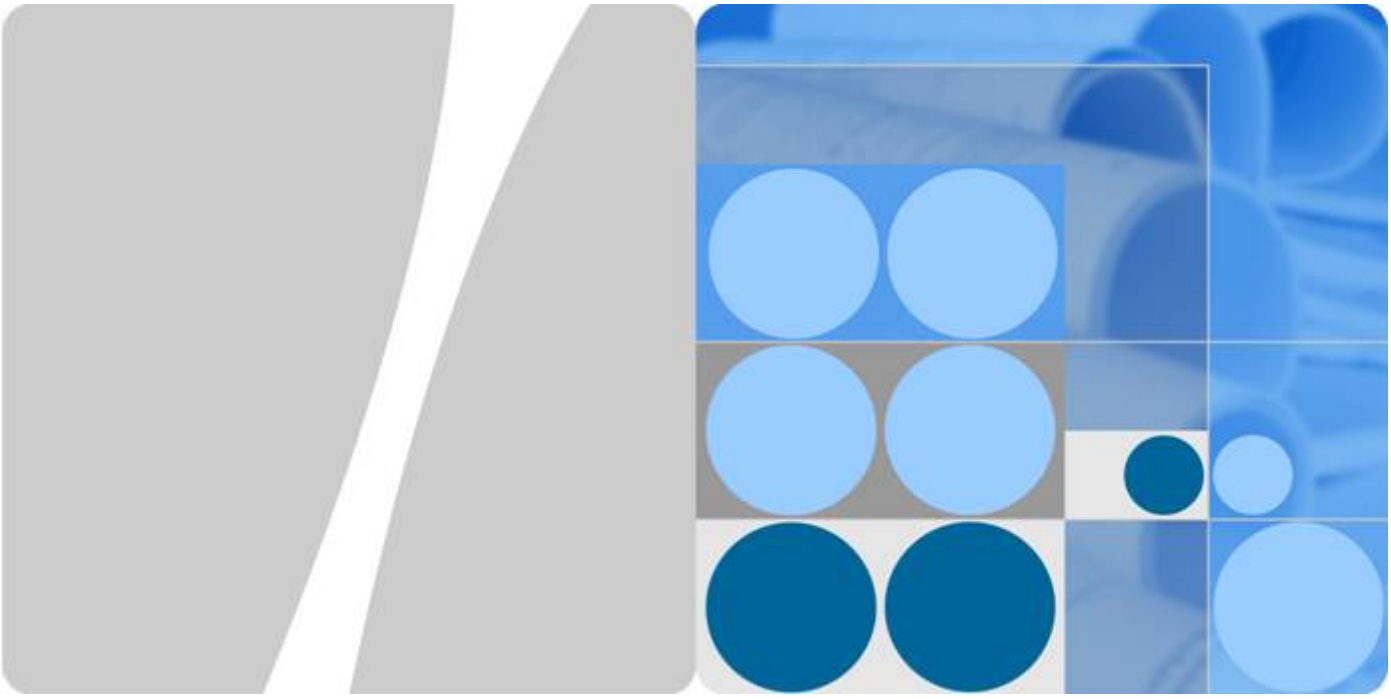


Distributed Message Service

API Reference

Issue 05

Date 2019-03-14

HUAWEI TECHNOLOGIES CO., LTD.

HUAWEI

Trademarks and Permissions


and other Huawei trademarks are trademarks of Huawei Technologies Co., Ltd.
All other trademarks and trade names mentioned in this document are the property of their respective holders.

Notice

The purchased products, services and features are stipulated by the contract made between Huawei and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

Contents

1 Before You Start

1.1 Overview

Welcome to *Distributed Message Service API Reference*. Distributed Message Service (DMS) is a message middleware service using the distributed, high-availability clustering technology. It provides reliable, scalable, and fully managed queues for sending, receiving, and storing messages.

This document describes functions, syntax, parameters, and examples of the application programming interfaces (APIs) of Distributed Message Service (DMS).

NOTICE

DMS is continuously upgraded with new functions, and the existing APIs are inevitably adjusted. For example, new response parameters are added.

To reduce the impact of API changes, DMS is backward compatible with existing APIs. When using DMS, you should accept and ignore unused parameters and parameter values in JSON responses.

1.2 API Calling

DMS supports Representational State Transfer (REST) APIs, allowing you to call APIs using HTTPS. For details about API calling, see **Calling APIs**.

1.3 Endpoints

An endpoint is the **request address** for calling an API. Endpoints vary depending on services and regions. For the endpoints of different services, see **Regions and Endpoints**.

1.4 Constraints

- The number of queues that you can create is determined by your quota. For details, see **Service Quota**.
- For more constraints, see the API description.

1.5 Concepts

- Account

  An account is created upon successful registration and has full access permissions for all of its cloud services and resources. It can be used to reset user passwords and grant user permissions. The account is a payment entity and should not be used directly to perform routine management. For security purposes, create IAM users and grant them permissions for routine management.

- IAM user

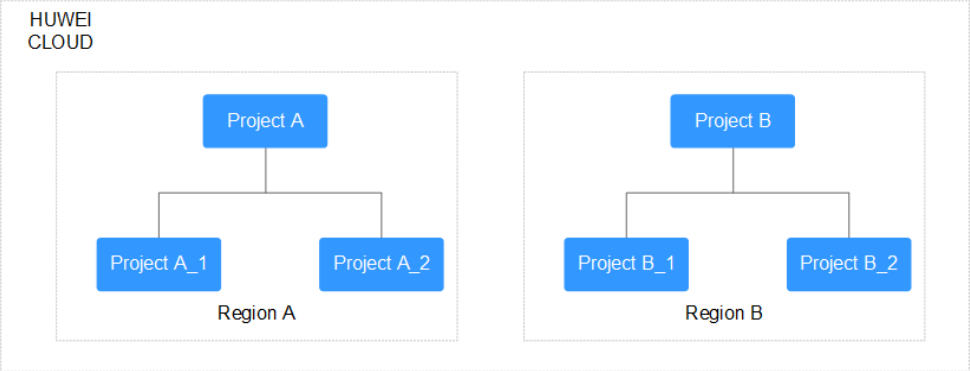
  An IAM user is created using an account to use cloud services. Each IAM user has its own identity credentials (password and access keys).

  The account name, username, and password will be required for API authentication.

- Regions are geographic areas isolated from each other. Resources are region-specific and cannot be used across regions through internal network connections. Deploying cloud resources in different regions can better suit certain user requirements or comply with local laws or regulations.
- An AZ contains one or more physical data centers. Each AZ has independent cooling, fire extinguishing, moisture-proof, and electricity facilities. Within an AZ, computing, network, storage, and other resources are logically divided into multiple clusters. AZs within a region are interconnected using high-speed optical fibers to support cross-AZ high-availability systems.
- Project

  Projects group and isolate resources (including compute, storage, and network resources) across physical regions. A default project is provided for each region, and subprojects can be created under each default project. Users can be granted permissions to access all resources in a specific project. For more refined access control, create subprojects under a project and purchase resources in the subprojects. Users can then be assigned permissions to access only specific resources in the subprojects.

Figure 1-1 Project isolating model



- Enterprise project

  Enterprise projects group and manage resources across regions. Resources in enterprise projects are logically isolated from each other. An enterprise project can contain resources in multiple regions, and resources can be directly transferred between enterprise projects.

  For details about how to obtain enterprise project IDs and features, see the **Enterprise Management User Guide**.

2 API Overview

Table 2-1 Queue and message management APIs

API	Description
Creating a Queue	Creates a standard queue.
Viewing All Queues	Views all queues.
Viewing a Queue	Views a specified queue.
Deleting a Queue	Deletes a specified queue.
Creating a Consumer Group	Creates one or more consumer groups for a specified queue.
Viewing All Consumer Groups of a Specified Queue	Views all consumer groups of a specified queue.
Deleting a Consumer Group	Deletes a specified consumer group.
Sending Messages to a Queue	Sends messages to a specified queue. Multiple messages can be sent at a time.
Consuming Messages	Consumes messages from a queue. Multiple messages can be consumed at a time. The load of messages consumed each time cannot exceed 512 KB.
Acknowledging Consumption of Specified Messages	Acknowledges consumption of specified messages.
Viewing Quotas	Views the quota of the current project.
Consuming Dead Letter Messages	Consumes dead letter messages from a queue. Multiple messages can be consumed at a time. The load of messages consumed each time cannot exceed 512 KB.

API	Description
Acknowledging Consumption of Specified Dead Letter Messages	Acknowledges consumption of specified dead letter messages.

3 Calling APIs

3.1 Making an API Request

This section describes the structure of a REST API request, and uses the IAM API for **obtaining a user token** as an example to demonstrate how to call an API. The obtained token can then be used to authenticate the calling of other APIs.

Request URI

A request URI is in the following format:

{URI-scheme} :// {Endpoint} / {resource-path} ? {query-string}

Although a request URI is included in the request header, most programming languages or frameworks require the request URI to be transmitted separately.

- **URI-scheme**:

  Protocol used to transmit requests. All APIs use HTTPS.

- **Endpoint**:

  Domain name or IP address of the server bearing the REST service. The endpoint varies between services in different regions. It can be obtained from **Regions and Endpoints**.

  For example, the endpoint of IAM in region **CN North-Beijing1** is **iam.cn-north-1.myhuaweicloud.com**.

- **resource-path**:

  Access path of an API for performing a specified operation. Obtain the value from the URI of the API. For example, the **resource-path** of the API used to **obtain a user token** is **/v3/auth/tokens**.

- **query-string**:

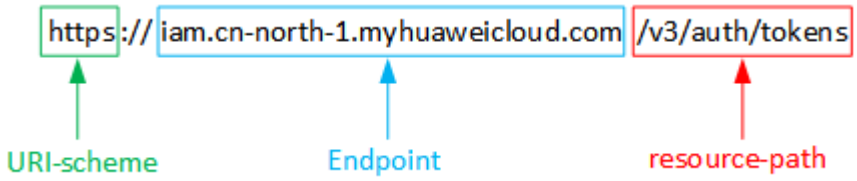
  Query parameter, which is optional. Ensure that a question mark (?) is included before a query parameter that is in the format of "*Parameter name=Parameter value*". For example, **? limit=10** indicates that a maximum of 10 data records will be displayed.

For example, to obtain an IAM token in the **CN North-Beijing1** region, obtain the endpoint of IAM **(iam.cn-north-1.myhuaweicloud.com)** for this region and the

resource-path (/v3/auth/tokens) in the URI of the API used to **obtain a user token**. Then, construct the URI as follows:

```
https://iam.cn-north-1.myhuaweicloud.com/v3/auth/tokens
```

Figure 3-1 Example URI



NOTE

To simplify the URI display in this document, each API is provided only with a **resource-path** and a request method. The **URI-scheme** of all APIs is **HTTPS**, and the endpoints of all APIs in the same region are identical.

Request Methods

The HTTP protocol defines the following request methods that can be used to send a request to the server:

- **GET**: requests the server to return specified resources.
- **PUT**: requests the server to update specified resources.
- **POST**: requests the server to add resources or perform special operations.
- **DELETE**: requests the server to delete specified resources, for example, an object.
- **HEAD**: requests a server resource header.
- **PATCH**: requests the server to update partial content of a specified resource. If the resource does not exist, a new resource will be created.

For example, in the case of the API used to **obtain a user token**, the request method is POST. The request is as follows:

```
POST https://iam.cn-north-1.myhuaweicloud.com/v3/auth/tokens
```

Request Header

You can also add additional header fields to a request, such as the fields required by a specified URI or HTTP method. For example, to request for the authentication information, add **Content-Type**, which specifies the request body type.

Common request headers are as follows:

- **Content-Type**: specifies the request body type or format. This field is mandatory and its default value is **application/json**. Other values of this field will be provided for specific APIs if any.
- **X-Auth-Token**: specifies a user token only for token-based API authentication. The user token is a response to the API used to **obtain a user token**. This API is the only one that does not require authentication.

> **NOTE**
>
> In addition to supporting token-based authentication, DMS APIs also support authentication using access key ID/secret access key (AK/SK). During AK/SK-based authentication, an SDK is used to sign the request, and the **Authorization** (signature information) and **X-Sdk-Date** (time when the request is sent) header fields are automatically added to the request.
>
> For more information, see **AK/SK-based Authentication**.

The API used to **obtain a user token** does not require authentication. Therefore, only the **Content-Type** field needs to be added to requests for calling the API. An example of such requests is as follows:

```
POST https://iam.cn-north-1.myhuaweicloud.com/v3/auth/tokens
Content-Type: application/json
```

Request Body

The body of a request is often sent in a structured format as specified in the **Content-Type** header field. The request body transfers content except the request header.

The request body varies between APIs. Some APIs do not require the request body, such as the APIs requested using the GET and DELETE methods.

In the case of the API used to **obtain a user token**, the request parameters and parameter description can be obtained from the API request. The following provides an example request with a body included. Replace *username*, *domainname*, ******** (login password), and *xxxxxxxxxxxxxxxxx* (project ID, such as cn-north-1) with the actual values. Obtain the project ID from the **Region** column of **Regions and Endpoints**.

> **NOTE**
>
> The **scope** parameter specifies where a token takes effect. You can set the **scope** to an account or a project under an account. In the following example, the token takes effect only for the resources in a specified project. For more information about this API, see **Obtaining a User Token**.

```
POST https://{{endpoint}}/v3/auth/tokens
Content-Type: application/json

{
    "auth": {
        "identity": {
            "methods": [
                "password"
            ],
            "password": {
                "user": {
                    "name": "username",
                    "password": "********",
                    "domain": {
                        "name": "domainname"
                    }
                }
            }
        },
        "scope": {
            "project": {
                "name": "xxxxxxxxxxxxxxxxx"
            }
        }
```



```
    }
}
```

If all data required by a request is available, you can send the request to call the API through **curl**, **Postman**, or coding. In the response to the API used to obtain a user token, **x-subject-token** is the desired user token. This token can then be used to authenticate the calling of other APIs.

3.2 Authentication

Requests for calling an API can be authenticated using either of the following methods:

- Token-based authentication: Requests are authenticated using a token.
- AK/SK-based authentication: Requests are authenticated by encrypting the request body using an AK/SK pair. AK/SK-based authentication is recommended because it provides higher security than token authentication.

Token-based Authentication

NOTE

The validity period of a token is 24 hours. When using a token for authentication, cache it to prevent frequently calling the IAM API used to obtain a user token.

A token specifies temporary permissions in a computer system. During API authentication using a token, the token is added to requests to get permissions for calling the API.

You can obtain a token by calling the API described in **Obtaining a User Token**. DMS APIs can be called only by using a project-level token. To call the API used to **obtain a user token**, set **auth.scope** to **project** in the request body as follows:

```
{
    "auth": {
        "identity": {
            "methods": [
                "password"
            ],
            "password": {
                "user": {
                    "name": "username",
                    "password": "********",
                    "domain": {
                        "name": "domainname"
                    }
                }
            }
        },
        "scope": {
            "project": {
                "name": "xxxxxxxx"
            }
        }
    }
}
```

After a token is obtained, the **X-Auth-Token** header field must be added to requests to specify the token when calling other APIs. For example, if the token is **ABCDEFJ....**, **X-Auth-Token: ABCDEFJ....** can be added to a request as follows:

```
POST https://{{endpoint}}/v3/auth/projects
Content-Type: application/json
X-Auth-Token: ABCDEFJ....
```

AK/SK-based Authentication

NOTE

AK/SK-based authentication supports API requests with a body not larger than 12 MB. For API requests with a larger body, token-based authentication is recommended.

In AK/SK-based authentication, AK/SK is used to sign requests and the signature is then added to the requests for authentication.

- AK: access key ID, which is a unique identifier used in conjunction with a secret access key to sign requests cryptographically.
- SK: secret access key used in conjunction with an AK to sign requests cryptographically. It identifies a request sender and prevents the request from being modified.

In AK/SK-based authentication, you can use an AK/SK to sign requests based on the signature algorithm or use the signing SDK to sign requests. For details about how to sign requests and use the signing SDK, see **API Request Signing Guide**.

NOTICE

The signing SDK is only used for signing requests and is different from the SDKs provided by services.

3.3 Response

Status Code

After sending a request, you will receive a response, including the status code, response header, and response body.

A status code is a group of digits, ranging from 1xx to 5xx. It indicates the status of a request. For more information, see **Status Code**.

For example, if status code **201** is returned for calling the API used to **obtain a user token**, the request is successful.

Response Header

Similar to a request, a response also has a header, for example, **Content-type**.

Figure 3-2 shows the response header for the API of **obtaining a user token**, in which **x-subject-token** is the desired user token. This token can then be used to authenticate the calling of other APIs.

Figure 3-2 Header fields of the response to the request for obtaining a user token

(Optional) Response Body

The body of a response is often returned in structured format as specified in the **Content-Type** header field. The response body transfers content except the response header.

The following is part of the response body for the API used to **obtain a user token**.

```
{
    "token": {
        "expires_at": "2019-02-13T06:52:13.855000Z",
        "methods": [
            "password"
        ],
        "catalog": [
            {
                "endpoints": [
                    {
                        "region_id": "XXXXXX",
......
```

If an error occurs during API calling, an error code and a message will be displayed. The following shows an error response body.

```
{
    "error_msg": "The format of message is error",
    "error_code": "AS.0001"
}
```

In the response body, **error_code** is an error code, and **error_msg** provides information about the error.

4 Getting Started

Scenarios

This section describes how to call an API to create a standard queue to store messages.

For details on how to call APIs, see **Calling APIs**.

Prerequisites

- IAM endpoint obtained from **Regions and Endpoints**.
- DMS endpoint obtained from **Regions and Endpoints**.

Creating a Queue

The following is an example request for creating a first-in-first-out (FIFO) queue:

```
{
  "name" : "queue-001",
  "queue_mode" : "FIFO",
  "redrive_policy" : "enable",
  "max_consume_count" : 3
}
```

- **name**: unique name of the queue
- **queue_mode**: type of the queue. Options:
  - **NORMAL**: Standard queue, which supports high concurrency performance but cannot guarantee that messages are retrieved in the exact sequence as how they are received.
  - **FIFO**: FIFO queue, which guarantees that messages are retrieved in the exact sequence as how they are received.
  - **KAFKA_HA**: High-reliability Kafka queue. All message replicas are flushed to a disk synchronously, ensuring message reliability.
  - **KAFKA_HT**: High-throughput Kafka queue. All message replicas are flushed to a disk asynchronously, ensuring high performance.
- **redrive_policy**: whether to enable dead letter messages. Options:
  - **enable**
  - **disable**

- **max_consume_count**: maximum number of allowed message consumption failures. When a message fails to be consumed after the number of consumption attempts of this message reaches this value, DMS stores this message into the dead letter queue.

For details about the parameters, see **Creating a Queue**.

5 APIs for Managing Queues and Messages

5.1 Creating a Queue

Function

By default, a maximum of 30 queues can be created for a project.

NOTE

When a queue is created, it takes one to three seconds to initialize the queue. Therefore, if operations such as message production, message consumption, queue details query, consumer group creation, and queue deletion are performed immediately after a queue is created, the operations may fail. You are advised to perform these operations three seconds after creating the queue.

URI

POST /v1.0/{project_id}/queues

Table 5-1 describes the parameter of this API.

Table 5-1 Parameter

Parameter	Type	Mandatory	Description
project_id	String	Yes	Indicates the ID of a project.

Request

Request parameters

Table 5-2 describes the parameters.

Table 5-2 Request parameters

Parameter	Type	Mandatory	Description
name	String	Yes	Indicates the unique name of a queue. The value is a string of 1 to 64 characters that contain letters, digits, hyphens (-), and underscores (_). The name cannot be modified once specified.
queue_mode	String	No	Indicates the queue type. Options: ● **NORMAL**: Standard queue, which supports high concurrency performance but cannot guarantee that messages are retrieved in the exact sequence as how they are received. ● **FIFO**: First-in-first-out (FIFO) queue, which guarantees that messages are retrieved in the exact sequence as how they are received. ● **KAFKA_HA**: High-reliability Kafka queue. All message replicas are flushed to a disk synchronously, ensuring message reliability. ● **KAFKA_HT**: High-throughput Kafka queue. All message replicas are flushed to a disk asynchronously, ensuring high performance. The default value is **NORMAL**.
description	String	No	Indicates the basic information about a queue. The value is a string of a maximum of 160 characters and cannot contain the angle brackets (<>).

Parameter	Type	Mandatory	Description
redrive_policy	String	No	This parameter is valid only when queue_mode is set to **NORMAL** or **FIFO**. This parameter specifies whether to enable dead letter messages. Dead letter messages are messages that cannot be normally consumed. When a message fails to be consumed after the number of consumption attempts of this message reaches this value, DMS stores this message into the dead letter queue. This message will be retained in the deal letter queue for 72 hours. During this period, consumers can consume the dead letter message. Dead letter messages can be consumed only by the consumer group that generated these dead letter messages. Dead letter messages of a FIFO queue are stored and consumed based on the FIFO sequence. Options: • **enable** • **disable** The default value is **disable**.
max_consume_count	Integer	No	This parameter is mandatory only when **redrive_policy** is set to **enable**. Indicates the maximum number of allowed message consumption failures. When a message fails to be consumed after the number of consumption attempts of this message reaches this value, DMS stores this message into the dead letter queue. Value range: 1–100.
retention_hours	Integer	No	Indicates the hours of storing messages in the Kafka queue. This parameter is valid only when **queue_mode** is set to **KAFKA_HA** or **KAFKA_HT**. Value range: 1–72.

Example request

Creating a FIFO queue:


```
{
  "name" : "queue-001",
  "description" : "This is a FIFO queue.",
  "queue_mode" : "FIFO",
  "redrive_policy" : "enable",
  "max_consume_count" : 3
}
```

Creating a Kafka queue:

```
{
  "name" : "queue-002",
  "description" : "This is a Kafka queue.",
  "queue_mode" : "KAFKA_HA",
  "retention_hours" : 36
}
```

Response

Response parameters

Table 5-3 describes the response parameters.

Table 5-3 Response parameters

Parameter	Type	Description
id	String	Indicates the queue ID.
name	String	Indicates the name of a queue.
kafka_topic	String	This parameter is returned only for Kafka queues. Indicates the Kafka topic ID when Kafka SDK is used.

Example response

Creating a FIFO queue:

```
{
    "id": "9bf46390-38a2-462d-b392-4d5b2d519c55",
    "name": "queue_001"
}
```

Creating a Kafka queue:

```
{
  "id" : "3ec7a4a2-541b-430a-9c2b-77fa4b64ed8",
  "name" : "queue_002",
  "kafka_topic" : "k-fdc60cfe407a4b2a96a498efda55c785-3ec7a4a2-541b-430a-9c2b-77fa4b64ed8"
}
```

Status Code

Table 5-4 lists the status code indicating that the operation is successful. For details about the status codes indicating that the operation fails, see **Status Code**.

Table 5-4 Status code

Status Code	Description
201	The queue is created successfully.

5.2 Viewing All Queues

Function

This API is used to view all queues.

URI

GET /v1.0/{project_id}/queues?include_deadletter={include_deadletter}

Table 5-5 describes the parameters of this API.

Table 5-5 Parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Indicates the project ID.
include_deadletter	Boolean	No	Indicates whether to list dead letter parameters in the response message. Options: • **true**: Include dead letter messages. • **false**: Do not include dead letter messages. The default value is **false**. Kafka queues do not support dead letter messages. This parameter is invalid for Kafka queues.

Example

```
GET v1.0/b78a90ae2a134b4b8b2ba30acab4e23a/queues?&include_deadletter=true
```

Request

Request parameters

None.

Example request

None.

Response

Response parameters

Table 5-6 and Table 5-7 describe the response parameters.

Table 5-6 Response parameters

Parameter	Type	Description
total	Integer	Indicates the total number of queues that belong to the tenant.
queues	Array	Indicates the total number of all queue arrays that belong to the tenant.

Table 5-7 queues parameters description

Parameter	Type	Description
id	String	Indicates the queue ID.
name	String	Indicates the queue name.
created	String	Indicates the time when a queue is created.
description	String	Indicates the basic information about a queue.
queue_mode	String	Indicates the queue type.
reservation	Integer	Indicates the retention period (unit: min) of a message in a queue.
max_msg_size_byte	Integer	Indicates the maximum message size (unit: byte) that is allowed in a queue.
produced_messages	Integer	Indicates the total number of messages in a queue.
redrive_policy	String	Indicates whether to enable dead letter messages. This parameter is displayed only when **include_deadletter** is set to **true**. Options: • **enable** • **disable**

Parameter	Type	Description
max_consume_count	Integer	Indicates the maximum number of allowed message consumption failures. When a message fails to be consumed after the number of consumption attempts of this message reaches this value, DMS stores this message into the dead letter queue. This parameter is displayed only when **include_deadletter** is set to **true**.
group_count	Integer	Indicates the number of consumption groups in a queue.
eff_date	String	Indicates the time when a queue is created.

Example response

```
{
  "queues" : [{
     "id" : "ef808d2d-58c2-4a36-9e58-d018b2193f80",
     "name" : "aaa_fifo_525",
     "description" : "test_fifo_detail",
     "queue_mode" : "NORMAL",
     "reservation" : 4320,
     "created" : 1495701557000,
     "max_msg_size_byte" : 524288,
     "produced_messages" : 1,
     "redrive_policy" : "enable",
     "max_consume_count" : 3,
     "eff_date": 1495701557000,
     "group_count" : 0
   }, {
     "id" : "bc0ac1ec-a4d6-4490-84cb-9d475f1ec3c5",
     "name" : "aaa_normal_525",
     "description" : "test",
     "queue_mode" : "NORMAL",
     "reservation" : 4320,
     "created" : 1495701490000,
     "max_msg_size_byte" : 524288,
     "produced_messages" : 0,
     "redrive_policy" : "enable",
     "max_consume_count" : 3,
     "eff_date": 1495701490000,
     "group_count" : 0
   }, {
     "id" : "1aaf34d0-7bb0-43be-9b71-f4b719d7ca47",
     "name" : "queue-normal",
     "description" : null,
     "queue_mode" : "NORMAL",
     "reservation" : 4320,
     "created" : 1495447342000,
     "max_msg_size_byte" : 524288,
     "produced_messages" : 2,
     "redrive_policy" : "enable",
     "max_consume_count" : 3,
     "eff_date": 1495447342000,
     "group_count" : 0
   }, {
     "id" : "f685ed59-43f4-4cf9-b609-7f333820d72d",
     "name" : "queue-835807102",
     "description" : "",
     "reservation" : 2160,
```



```
    "created" : 1517379348000,
    "queue_mode" : "KAFKA_HA",
    "max_msg_size_byte" : 524288,
    "produced_messages" : 0,
    "eff_date": 1517379348000,
    "group_count" : 0
   }
 ],
 "total" : 4
}
```


Status Code

Table 5-8 lists the status code indicating that the operation is successful. For details about the status codes indicating that the operation fails, see **Status Code**.

Table 5-8 Status code

Status Code	Description
200	The information is obtained successfully.

5.3 Viewing a Queue

Function

This API is used to view a specified queue.

URI

GET /v1.0/{project_id}/queues/{queue_id}?include_deadletter={include_deadletter}

Table 5-9 describes the parameters of this API.

Table 5-9 Parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Indicates the ID of a project.
queue_id	String	Yes	Indicates the ID of the queue to be viewed.

Parameter	Type	Mandatory	Description
include_deadletter	Boolean	No	Indicates whether to list dead letter parameters in the response message. Options: • **true**: Include dead letter messages. • **false**: Do not include dead letter messages. The default value is **false**. Kafka queues do not support dead letter messages. This parameter is invalid for Kafka queues.

Example

```
GET v1.0/b78a90ae2a134b4b8b2ba30acab4e23a/queues/075ae7da-6ce5-4966-940c-17c19fb5175e?include_deadletter=true
```

Request

Request parameters

None.

Example request

None.

Response

Response parameters

Table 5-10 describes the response parameters.

Table 5-10 Response parameters

Parameter	Type	Description
id	String	Indicates the queue ID.
name	String	Indicates the queue name.
created	String	Indicates the time when a queue is created.
description	String	Indicates the basic information about a queue.
queue_mode	String	Indicates the queue type.
reservation	Integer	Indicates the retention period (unit: min) of a message in a queue.
max_msg_size_byte	Integer	Indicates the maximum message size (unit: byte) that is allowed in a queue.

Parameter	Type	Description
produced_messages	Integer	Indicates the total number of messages in a queue.
redrive_policy	String	Indicates whether to enable dead letter messages. This parameter is displayed only when **include_deadletter** is set to **true**. Options: • **enable** • **disable**
max_consume_count	Integer	Indicates the maximum number of allowed message consumption failures. When a message fails to be consumed after the number of consumption attempts of this message reaches this value, DMS stores this message into the dead letter queue. This parameter is displayed only when **include_deadletter** is set to **true**.
group_count	Integer	Indicates the number of consumption groups in a queue.
kafka_topic	String	This parameter is returned only for Kafka queues.
eff_date	String	Indicates the time when a queue is created.

Example response

```
{
    "id": "0611d466-a327-4b7b-8034-f84a0f6a6f42",
    "name": "queue-001",
    "description": "This is a FIFO queue.",
    "reservation": 4320,
    "created": 1558691803000,
    "queue_mode": "FIFO",
    "max_msg_size_byte": 524288,
    "produced_messages": 14,
    "eff_date": 1558691803000,
    "group_count": 1
}
```

Status Code

Table 5-11 lists the status code indicating that the operation is successful. For details about the status codes indicating that the operation fails, see **Status Code**.

Table 5-11 Status code

Status Code	Description
200	The information is obtained successfully.

5.4 Deleting a Queue

Function

This API is used to delete a specified queue.

URI

DELETE /v1.0/{project_id}/queues/{queue_id}

Table 5-12 describes the parameters of this API.

Table 5-12 Parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Indicates the ID of a project.
queue_id	String	Yes	Indicates the ID of the queue to be deleted.

Request

Request parameters

None.

Example request

None.

Response

Response parameters

None.

Example response

None.

Status Code

Table 5-13 lists the status code indicating that the operation is successful. For details about the status codes indicating that the operation fails, see **Status Code**.

Table 5-13 Status code

Status Code	Description
204	The queue is deleted successfully.

5.5 Creating a Consumer Group

Function

This API is used to create a consumer group.

Multiple consumer groups can be created for a queue at a time.

NOTE

After you create a consumer group, it takes 1s to 3s to initialize the system. If you operate the queue immediately after creating the consumer group, the consumption may fail. You are advised to perform these operations three seconds after creating the queue.

URI

POST /v1.0/{project_id}/queues/{queue_id}/groups

Table 5-14 describes the parameters of this API.

Table 5-14 Parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Indicates the ID of a project.
queue_id	String	Yes	Indicates the ID of a queue.

Request

Request parameters

Table 5-15 describes the request parameters.

Table 5-15 Request parameters

Parameter	Type	Mandatory	Description
groups	Array	Yes	Indicates the consumer group information. A maximum of three consumer groups can be created for a queue. If there are more than three consumer groups in the request, the request will fail and consumer groups cannot be created.

Table 5-16 groups parameter description

Parameter	Type	Mandatory	Description
name	String	Yes	Indicates the name of a consumer group. The value is a string of 1 to 32 characters that contain letters, digits, hyphens (-), and underscores (_).

Example request

```
{
  "groups" : [{
     "name" : "group-aa"
   }
  ]
}
```

Response

Response parameters

Table 5-17 describes the response parameters.

Table 5-17 Response parameters

Parameter	Type	Description
id	String	Indicates the consumer group ID.
name	String	Indicates the name of a consumer group.

Example response

```
{
  "groups" : [{
```

```
    "id" : "g-02fb1974-9be1-4eee-8448-ed2d3e89884a",
    "name" : "group-aa"
  }
 ]
}
```

Status Code

Table 5-18 lists the status code indicating that the operation is successful. For details about the status codes indicating that the operation fails, see Status Code.

Table 5-18 Status code

Status Code	Description
201	The consumer group is created successfully.

5.6 Viewing All Consumer Groups of a Specified Queue

Function

This API is used to view all consumer groups of a specified queue.

URI

GET /v1.0/{project_id}/queues/{queue_id}/groups?include_deadletter={include_deadletter}&include_messages_num={boolean}&page_size={page_size}&current_page={current_page}

Table 5-19 describes the parameters of this API.

Table 5-19 Parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Indicates the project ID.
queue_id	String	Yes	Indicates the queue ID.
include_deadletter	Boolean	No	Indicates whether to list dead letter parameters in the response message. The default value is **false**.
include_messages_num	Boolean	No	Indicates whether to query the details of consumer groups. Default value: **true**. If this parameter is set to false, consumption details of a consumer group are not queried, and the API responses in a short time.

Parameter	Type	Mandatory	Description
page_size	Integer	No	Indicates the number of consumer groups displayed on each page. If **page_size** and **current_page** are not set to a valid value at the same time, consumer groups displayed on all pages are queried by default.
current_page	Integer	No	Indicates the number of a page on which consumer groups are to be queried. If **page_size** and **current_page** are not set to a valid value at the same time, consumer groups displayed on all pages are queried by default.

Example

```
GET v1.0/b78a90ae2a134b4b8b2ba30acab4e23a/queues/075ae7da-6ce5-4966-940c-17c19fb5175e/groups?include_deadletter=true
```

Request

Request parameters

None.

Example request

None.

Response

Response parameters

Table 5-20 and **Table 5-21** describe the response parameters.

Table 5-20 Response parameters

Parameter	Type	Description
queue_id	String	Indicates the queue ID.
queue_name	String	Indicates the queue name.
groups	Array	Indicates the consumer group list.

Parameter	Type	Description
redrive_policy	String	Indicates whether to enable dead letter messages. This parameter is displayed only when **include_deadletter** is set to **true**. Options: • **enable** • **disable**

Table 5-21 groups parameter description

Parameter	Type	Description
id	String	Indicates the consumer group ID.
name	String	Indicates the name of a consumer group.
produced_messages	Integer	Indicates the total number of messages (not including the messages that have expired and been deleted) in a queue.
consumed_messages	Integer	Indicates the total number of messages that are successfully consumed.
available_messages	Integer	Indicates the accumulated number of normal messages available to the consumer group.
produced_deadletters	Integer	Indicates the total number of dead letter messages generated by the consumer group. This parameter is displayed only when **include_deadletter** is set to **true**.
available_deadletters	Integer	Indicates the accumulated number of dead letter messages not consumed in the consumer group. This parameter is displayed only when **include_deadletter** is set to **true**.

Example response

```
{
   "queue_name" : "queue-772289871",
   "groups" : [{
          "name" : "group-1690260950",
          "id" : "g-eb9305bb-5bec-4712-84ab-0a36fbe9c2c0",
          "consumed_messages" : 0,
          "available_messages" : 8,
          "produced_messages" : 10,
      }
   ],
   "redrive_policy" : "enable",
   "queue_id" : "f5b6dd28-08dd-4f0f-866c-2eadf6788163"
}
```


When **include_messages_num** is set to **false**:

```
{
   "queue_name" : "queue-586845368",
   "groups" : [{
         "name" : "group-364417183",
         "id" : "g-33d53064-2ab9-4acc-8566-3faa8c8578bf",
         "consumed_messages" : 0,
         "available_messages" : 0,
         "produced_messages" : 0,
      }, {
         "name" : "group-1722391629",
         "id" : "g-876fc3a2-e8c1-4a81-af3e-9ef68e3e46cf",
         "consumed_messages" : 0,
         "available_messages" : 0,
         "produced_messages" : 0,
      }
   ],
   "queue_id" : "e7e6d7f6-c555-470a-b9ee-3175e3408250"
}
```

Status Code

Table 5-22 lists the status code indicating that the operation is successful. For details about the status codes indicating that the operation fails, see **Status Code**.

Table 5-22 Status code

Status Code	Description
200	The information is obtained successfully.

5.7 Deleting a Consumer Group

Function

This API is used to delete a specified consumer group.

URI

DELETE /v1.0/{project_id}/queues/{queue_id}/groups/{group_id}

Table 5-23 describes the parameters of this API.

Table 5-23 Parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Indicates the ID of a project.
queue_id	String	Yes	Indicates the queue ID.
group_id	String	Yes	Indicates the ID of the consumer group to be deleted.

Request

Request parameters

None.

Example request

None.

Response

Response parameters

None.

Example response

None.

Status Code

Table 5-24 lists the status code indicating that the operation is successful. For details about the status codes indicating that the operation fails, see **Status Code**.

Table 5-24 Status code

Status Code	Description
204	The consumer group is deleted successfully.

5.8 Sending Messages to a Queue

Function

This API is used to send messages to a queue. Multiple messages can be sent at a time. The following requirements must be met:

- A maximum of 10 messages can be sent at a time.
- The aggregated size of messages sent at a time cannot exceed 512 KB.
- In Kafka queues, messages are retained for 1 to 72 hours, depending on what you choose when creating a queue. In the other queues, messages are retained for at least 72 hours and will be deleted after expiry.

URI

POST /v1.0/{project_id}/queues/{queue_id}/messages

Table 5-25 describes the parameters of this API.

Table 5-25 Parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Indicates the project ID.
queue_id	String	Yes	Indicates the queue ID.

Request

Request parameters

Table 5-26 and Table 5-27 list the parameter description.

Table 5-26 Request parameters

Parameter	Type	Mandatory	Description
messages	Array	Yes	Indicates the message list.
returnId	Boolean	No	Indicates whether to return a message ID after a message is sent successfully. The default value is **false**. A message ID is returned only if the value is set to **true**.

Table 5-27 messages parameter description

Parameter	Type	Mandatory	Description
body	JSON	Yes	Indicates the message body.
attributes	JSON object	No	Indicates the list of attributes, including attribute names and values. The attribute name must be unique for a message.
tags	JSON object	No	Indicates the message label, which is used to identify message types. You can use message labels to filter the messages you want to retrieve from the chosen queue. A message label is 1 to 64 characters long. Only letters, digits, hyphens (-), and underscores (_) are allowed. A maximum of 3 labels can be added for a message.

Parameter	Type	Mandatory	Description
delayTime	Long	No	Indicates the delay time of a delay message delivery. Amount of time to delay delivery of all messages added to this queue. Value range: 0–604800000 Unit: ms If this parameter is not set or is set to **0**, there is no delivery delay. If this parameter is set to a floating point number, the integer value before the decimal point is automatically used. For example, if this parameter is set to **6000.9**, the value is automatically set to **6000**. Only FIFO and NORMAL queues support message delivery delay. If you enable delivery delay for messages in a Kafka queue, an error message "{"code":10540010, "message":"Invalid request format: kafka queue message could not have delayTime."}" is displayed.

Example request

```
{
   "messages" : [{
          "body" : "TEST11",
          "attributes" : {
              "attribute1" : "value1",
              "attribute2" : "value2"
          },
          "tags" : ["tag1", "tag2"],
           "delayTime":60000
       }, {
          "body" : {
              "foo" : "test02"
          },
          "attributes" : {
              "attribute1" : "value1",
              "attribute2" : "value2"
          },
          "tags" : ["tag1", "tag2"],
          "delayTime":10000
       }
   ],
   "returnId" : "true"
}
```

Response

Response parameters

Table 5-28 and Table 5-29 describe the response parameters.

Table 5-28 Response parameters

Parameter	Type	Description
message	Array	Indicates the message list.

Table 5-29 messages response parameter

Parameter	Type	Description
error	String	Indicates the error information.
error_code	Integer	Indicates the error code (if any).
state	Integer	Indicates the message sending status. **0**: Messages are successfully sent. **1**: Messages failed to be sent. The **error** and **error_code** parameters indicate the cause of failure.
id	String	Indicates the message ID.

Example response


```
{
  "messages" : [{
      "error" : null,
      "state" : 0,
      "id" :
"eyJ0b3BpYyI6InEtNjdjMDFiOTI5NDQxNDRhMTlkMmRhOTY4ZWYzNGE5MTItNGZhMWQ5YTQtNjRhNC00M
mYxLTk3MzAtZGU4NTFjMTU0Mjg2Iiwib2Zmc2V0IjoyMzQ4LCJwYXJ0aXRpb24iOjIsInRhcmdldFRvcGljIjpudWxs
fQ==",
      "error_code" : null
    },
    {
      "error" : null,
      "state" : 0,
      "id" : "jhGwdWEpnyrXMlauz72j+7cD8W9F4I2HAk6GyQFJCJMX6Va3W7KlA2IVCZ
+hYHFcKqA0n1DQLdKMCyGKvd0ZrQRfwHzjAabgYnWg2VCHtb12fJkzKMQB4JwwcyHvsPNffmFW6gxC4VVaT
4cHf8sLYzrZmES1fd36r5o9wpbpqOgi2I==",
      "error_code" : null
    }
  ]
}
```


Status Code

Table 5-30 lists the status code indicating that the operation is successful. For details about the status codes indicating that the operation fails, see **Status Code**.

Table 5-30 Status code

Status Code	Description
201	Messages are sent successfully.

5.9 Consuming Messages

Function

This API is used to consume messages in a specified queue. Multiple messages can be consumed at a time. The load of messages consumed each time cannot exceed 512 KB.

When there are only a few messages in a queue, the number of messages actually consumed at a time may be less than the message quantity specified in the consumption request. However, all messages in the queue will be eventually obtained by the message consumer after multiple rounds of consumption. If the queue is empty, no message will be returned to the consumer.

Once a consumer group specifies a message label, the consumer group must use the label for all subsequent retrievals. If a consumer group changes a label during the next retrieval, the next retrieval will fail.

URI

GET /v1.0/{project_id}/queues/{queue_id}/groups/{consumer_group_id}/messages?max_msgs={max_msgs}&time_wait={time_wait}&ack_wait={ack_wait}&tag={tag1}&tag={tag2}&tag_type={TYPE}

Table 5-31 describes the parameters of this API.

Table 5-31 Parameters

Parameter	Type	Mandatory	Description	Value Range
project_id	String	Yes	Indicates the ID of a project.	N/A
queue_id	String	Yes	Indicates the queue ID.	N/A
consumer_group_id	String	Yes	Indicates the consumer group ID. Obtain the consumer group ID from the response message in **Viewing All Consumer Groups of a Specified Queue**.	N/A

Parameter	Type	Mandatory	Description	Value Range
max_msgs	Integer	No	Indicates the number of consumable messages that can be obtained per time.	Value range: 1–10. Default Value: **10**.
time_wait	Integer	No	Indicates the amount of time that the API call can wait for a message to arrive in the empty queue before returning an empty response. If a message is available during the wait period, the call will return the message consumption result immediately. If no message is available until the wait period expires, the call will return an empty response after the wait period expires.	Value range: 1 to 60 seconds. Default value: **3s**. The default wait period is 3 seconds even if the API request does not carry the **time_wait** parameter or the **time_wait** parameter in the API request is left unspecified.
ack_wait	Integer	No	Indicates the timeout duration that the API call can wait for message consumption acknowledgement. The client needs to submit the message consumption acknowledgement within the specified time. If the message consumption is not acknowledged within this period of time, the system displays a message, indicating that message consumption acknowledgement has timed out or the handler is invalid. In this case, the system determines that the message fails to be consumed by default.	Value range: 15 to 300 seconds. Default value: **30s**. If this parameter is left unspecified or empty, the default value **30s** is used.

Parameter	Type	Mandatory	Description	Value Range
tag	String	No	Indicates the message label. You can use message labels to filter the messages you want to retrieve from the chosen queue.	The number of labels cannot exceed 3. A label contains a maximum of 64 characters.
tag_type	String	No	Indicates the message filter mode when multiple message labels are configured.	Options: • **and**: Only messages that match all labels are consumed. • **or**: Messages that match any one label are consumed. Default value: **or**.

Example

```
GET v1.0/b78a90ae2a134b4b8b2ba30acab4e23a/queues/075ae7da-6ce5-4966-940c-17c19fb5175e/groups/g-5ec247fd-d4a2-4d4f-9876-e4ff3280c461/messages?max_msgs=10&ack_wait=30&tag=tag1&tag=tag2&tag_type=or
```

Request

Request parameters

None.

Example request

None.

Response

Response parameters

Table 5-32 and Table 5-33 describe the response parameters.

Table 5-32 Response parameters

Parameter	Type	Description
message	JSON object	Indicates the message content.
handler	String	Indicates the message handler.

Table 5-33 message parameter description

Parameter	Type	Description
body	JSON	Indicates the message body.
attributes	JSON object	Indicates the list of attributes.
tags	JSON array	Indicates the message label.

Example response

```
[{
  "message" : {
   "body" : {
    "foo" : "123="
   },
   "attributes": {
       "attribute1": "value1",
       "attribute2": "value2"
        }
  },
  "tags":["tag1","tag2"],
  "handler" :
"eyJjZyI6Im15X2pzb25fZ3JvdXAiLCJjaSI6InJlc3QtY29uc3VtZXItYzNlNThiNjEtYzA0NC00NGJkLTkxM2ItZDgzNjljNmJhYTQxIiwiY291bnQiOjAsIm9mZnNldCI6MCwicCI6MCwidCI6InRlc3QyIn0="
 }, {
  "message" : {
   "body" : {
    "foo" : "123="
   },
   "attributes": {
       "attribute1": "value1",
       "attribute2": "value2"
        }
  },
  "tags":["tag1","tag2"],
  "handler" :
"eyJjZyI6Im15X2pzb25fZ3JvdXAiLCJjaSI6InJlc3QtY29uc3VtZXItYzNlNThiNjEtYzA0NC00NGJkLTkxM2ItZDgzNjljNmJhYTQxIiwiY291bnQiOjAsIm9mZnNldCI6MSwicCI6MCwidCI6InRlc3QyIn0="
 }
]
```

Status Code

Table 5-34 lists the status code indicating that the operation is successful. For details about the status codes indicating that the operation fails, see **Status Code**.

Table 5-34 Status code

Status Code	Description
200	The information is obtained successfully.

5.10 Acknowledging Consumption of Specified Messages

Function

This API is used to confirm the consumption of specified messages.

While a message is being consumed, it remains in the queue and cannot be consumed again within 30s. If the message fails to be consumed within 30s, it can be consumed again.

If a message is consumed, it cannot be consumed again by the consumer group. However, it is retained in the queue (unless the queue is deleted) and can be consumed by other consumer groups. The retention period is 72 hours by default and the message will be deleted 72 hours later.

After a batch of messages is consumed, consumers must acknowledge the message consumption in the exact order that the messages are consumed. DMS checks whether messages are successfully consumed in the same order. If a message has not been acknowledged as a consumed message or failed to be consumed, DMS stops checking and determines that all the subsequent messages fail to be consumed. Therefore, when a consumer fails to acknowledge the consumption of a message (in a batch of messages), you are advised to stop the consumer from consuming the rest of the messages, and acknowledge the consumption of messages that have been successfully consumed.

If the consumption of a message fails to be acknowledged, this message can be re-consumed and its consumption can also be acknowledged again. If dead letter messages are enabled and a message fails to be consumed for a preset number of times, the message will be sent to the dead letter queue and retained in the dead letter queue for a maximum of 72 hours. You can then consume the message from the dead letter queue.

URI

POST /v1.0/{project_id}/queues/{queue_id}/groups/{consumer_group_id}/ack

Table 5-35 describes the parameters of this API.

Table 5-35 Parameter description

Parameter	Type	Mandatory	Description
project_id	String	Yes	Indicates the ID of a project.
queue_id	String	Yes	Indicates the queue ID.
consumer_group_id	String	Yes	Indicates the consumer group ID.

Request

Request parameters

Table 5-36 and Table 5-37 list the parameter description.

Table 5-36 Request parameters

Parameter	Type	Mandatory	Description
message	Array	Yes	Indicates the message confirmation arrays.

Table 5-37 message parameter description

Parameter	Type	Mandatory	Description
handler	String	Yes	Indicates the ID returned during the consumption.
status	String	No	Indicates the message consumption status. The value can be **success** or **fail**.

Example request

```
{
  "message": [
    {
      "handler":
"eyJjb25zdW1lckdyb3VwIjoibXFzX2NvbnN1bWVyXzMiLCJjb25zdW1lckluc3RhbmNlIjoicmVzdC1jb25zdW1lci1hMWM5YTRlMy1mNTY5LTQyYTgtOTQ1Ni1hYmU0NDVmZjUxYzkiLCJjb3VudCI6MSwib2Zmc2V0IjowLCJvZmZzZXRJbmRleCI6LTEsInBhcnRpdGlvbiI6MiwidG9waWMiOiJxLWI3OGE5MGFlMmExMzRiNGI4YjJiYTMwYWNhYjRlMjNhLTA3NWFlN2RhLTZjZTUtNDk2Ni05NDBjLTE3YzE5ZmI1MTc1ZSJ9",
      "status": "success"
    }
  ]
}
```

Response

Response parameters

Table 5-38 describes the response parameters.

Table 5-38 Response parameters

Parameter	Type	Description
success	Integer	Indicates the number of messages that are successfully acknowledged. The value **N** indicates that the first *N* messages are successfully acknowledged.

Parameter	Type	Description
fail	Integer	Indicates the number of messages that failed to be acknowledged. The value **N** indicates that the last *N* messages failed to be acknowledged.

Example response

```
{
  "success": 1,
  "fail": 2
}
```

Status Code

Table 5-39 lists the status code indicating that the operation is successful. For details about the status codes indicating that the operation fails, see **Status Code**.

Table 5-39 Status code

Status Code	Description
200	The consumption of the message is successfully acknowledged.

5.11 Viewing Quotas

Function

This API is used to view the quota of the current project.

URI

GET /v1.0/{project_id}/quotas/dms

Table 5-40 describes the parameters of this API.

Table 5-40 Parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Indicates the ID of a project.

Request

Request parameters

None.

Example request

None.

Response

Response parameters

Table 5-41, **Table 5-42**, and **Table 5-43** describe the response parameters.

Table 5-41 Response parameters

Parameter	Type	Description
quotas	JSON	Indicates the quotas of a tenant.

Table 5-42 quotas parameter description

Parameter	Type	Description
resources	Array	Indicates the list of quotas.

Table 5-43 resources parameter description

Parameter	Type	Description
type	String	Indicates the name of a quota.
quota	Integer	Indicates the total quota.
used	Integer	Indicates the used quota.
min	Integer	Indicates the minimum value that a quota must reach.
max	Integer	Indicates the maximum value that a quota cannot exceed.

Example response

```
{
  "quotas" : {
    "resources" : [{
        "type" : "queue",
        "quota" : 30,
        "used" : 5,
        "min" : 0,
        "max" : 500
      }
    ]
  }
}
```

Status Code

Table 5-44 lists the status code indicating that the operation is successful. For details about the status codes indicating that the operation fails, see **Status Code**.

Table 5-44 Status code

Status Code	Description
200	The information is obtained successfully.

5.12 Consuming Dead Letter Messages

Function

This API is used to consume the dead letter messages generated by a specified consumer group. Multiple messages can be consumed at a time. The load of messages consumed each time cannot exceed 512 KB.

Only NORMAL and FIFO queues can consume dead letter messages and therefore can have the Dead Letter Message function enabled.

URI

GET /v1.0/{project_id}/queues/{queue_id}/groups/{consumer_group_id}/deadletters?max_msgs={max_msgs}&time_wait={time_wait}&ack_wait={ack_wait}

Table 5-45 describes the parameters of this API.

Table 5-45 Parameters

Parameter	Type	Mandatory	Description	Value Range
project_id	String	Yes	Indicates the ID of a project.	N/A
queue_id	String	Yes	Indicates the queue ID.	N/A
consumer_group_id	String	Yes	Indicates the consumer group ID. You can obtain the consumer group ID from the response message returned after you call the API to view all consumer groups of a specified queue. For details, see **Viewing All Consumer Groups of a Specified Queue**.	N/A

Parameter	Type	Mandatory	Description	Value Range
max_msgs	Integer	No	Indicates the number of consumable dead letter messages that can be obtained a time. **NOTE** The number of dead letter messages actually consumed at a time may be less than the message quantity specified in the consumption request. However, all dead letter messages in the queue will be eventually obtained by the consumer after multiple rounds of consumption.	Value range: 1–10. Default Value: **10**.
time_wait	Integer	No	Indicates the waiting time for reading messages when the number of dead letter messages that can be consumed by a consumer group is 0. If a dead letter message is available during the wait period, the message consumption result is returned immediately. If no dead letter message is available until the wait period expires, an empty response will be returned after the wait period ends.	Value range: 1 to 60 seconds. Default value: **3s**. The default wait period is 3 seconds even if the API request does not carry the **time_wait** parameter or the **time_wait** parameter in the API request is left unspecified.

Parameter	Type	Mandatory	Description	Value Range
ack_wait	Integer	No	Indicates the commit submission timeout interval. Acknowledgement within the time specified by this parameter is valid. Otherwise, the system displays a message indicating that the message acknowledgement times out or the handler is invalid.	Value range: 15 to 300 seconds. Default value: **30s**. If this parameter is left unspecified or empty, the default value **30s** is used.

Example

```
v1.0/b78a90ae2a134b4b8b2ba30acab4e23a/queues/075ae7da-6ce5-4966-940c-17c19fb5175e/groups/g-5ec247fd-d4a2-4d4f-9876-e4ff3280c461/deadletters?max_msgs=10&ack_wait=30
```

Request

Request parameters

None.

Example request

None.

Response

Response parameters

Table 5-46 and Table 5-47 describe the response parameters.

Table 5-46 Response parameters

Parameter	Type	Description
message	JSON object	Indicates the message content.
handler	String	Indicates the message handler.

Table 5-47 message parameter description

Parameter	Type	Description
body	JSON	Indicates the message body.
attributes	JSON object	Indicates the list of attributes.

Example response

```
[{
  "message" : {
   "body" : {
    "foo" : "123="
   },
   "attributes": {
       "attribute1": "value1",
       "attribute2": "value2"
       }
  },
  "handler" :
"eyJjZyI6Im15X2pzb25fZ3JvdXAiLCJjaSI6InJlc3QtY29uc3VtZXItYzNlNThiNjEtYzA0NC00NGJkLTkxM2ItZDgzNjljNmJhYTQxIiwiY291bnQiOjAsIm9mZnNldCI6MCwicCI6MCwidCI6InRlc3QyIn0="
 }, {
  "message" : {
   "body" : {
    "foo" : "123="
   },
   "attributes": {
       "attribute1": "value1",
       "attribute2": "value2"
       }
  },
  "handler" :
"eyJjZyI6Im15X2pzb25fZ3JvdXAiLCJjaSI6InJlc3QtY29uc3VtZXItYzNlNThiNjEtYzA0NC00NGJkLTkxM2ItZDgzNjljNmJhYTQxIiwiY291bnQiOjAsIm9mZnNldCI6MSwicCI6MCwidCI6InRlc3QyIn0="
 }
]
```

Status Code

Table 5-48 lists the status code indicating that the operation is successful. For details about the status codes indicating that the operation fails, see **Status Code**.

Table 5-48 Status code

Status Code	Description
200	The information is obtained successfully.

5.13 Acknowledging Consumption of Specified Dead Letter Messages

Function

This API is used to confirm the consumption of specified dead letter messages.

When a dead letter message is being consumed, it remains in the queue. It cannot be consumed again by the same consumer group within 30s since the start of the consumption. If consumption is not acknowledged within this period, DMS determines that this dead letter message fails to be consumed, and this dead letter message can be consumed again.

Once consumption is acknowledged, this dead letter message can no longer be consumed by the same consumer group. Dead letter messages remain in the queue for 72 hours (unless the consumer group is deleted) and will be deleted after this period.

After a batch of messages is consumed, consumers must acknowledge the message consumption in the exact order that the messages are consumed. DMS checks whether messages are successfully consumed in the same order. If a message has not been acknowledged as a consumed message or failed to be consumed, DMS stops checking and determines that all the subsequent messages fail to be consumed. Therefore, when a consumer fails to acknowledge the consumption of a message (in a batch of messages), you are advised to stop the consumer from consuming the rest of the messages, and acknowledge the consumption of messages that have been successfully consumed.

Only NORMAL and FIFO queues can consume dead letter messages and therefore can have the Dead Letter Message function enabled.

URI

POST /v1.0/{project_id}/queues/{queue_id}/groups/{consumer_group_id}/deadletters/ack

Table 5-49 describes the parameters of this API.

Table 5-49 Parameters

Parameter	Type	Mandatory	Description
project_id	String	Yes	Indicates the ID of a project.
queue_id	String	Yes	Indicates the queue ID.
consumer_group_id	String	Yes	Indicates the consumer group ID.

Request

Request parameters

Table 5-50 and **Table 5-51** list the parameter description.

Table 5-50 Request parameters

Parameter	Type	Mandatory	Description
message	Array	Yes	Indicates the message confirmation arrays.

Table 5-51 message parameter description

Parameter	Type	Mandatory	Description
handler	String	Yes	Indicates the ID returned during the consumption.
status	String	No	Indicates the message consumption status. The value can be **success** or **fail**.

Example request

```
{
  "message": [
    {
      "handler":
"eyJjb25zdW1lckdyb3VwIjoibXFzX2NvbnN1bWVyXzMiLCJjb25zdW1lckluc3RhbmNlIjoicmVzdC1jb25zdW1lci1hMWM5YTRlMy1mNTY5LTQyYTgtOTQ1Ni1hYmU0NDVmZjUxYzkiLCJjb3VudCI6MSwib2Zmc2V0IjowLCJvZmZzZXRJbmRleCI6LTEsInBhcnRpdGlvbiI6MiwidG9waWMiOiJxLWI3OGE5MGFlMmExMzRiNGI4YjJiYTMwYWNhYjRlMjNhLTA3NWFlN2RhLTZjZTUtNDk2Ni05NDBjLTE3YzE5ZmI1MTc1ZSJ9",
      "status": "success"
    }
  ]
}
```

Response

Response parameters

Table 5-52 describes the response parameters.

Table 5-52 Response parameters

Parameter	Type	Description
success	Integer	Indicates the number of dead letter messages that are successfully acknowledged. The value **N** indicates that the first *N* dead letter messages are successfully acknowledged.
fail	Integer	Indicates the number of dead letter messages that failed to be acknowledged. The value **N** indicates that the first *N* dead letter messages failed to be acknowledged.

Example response

```
{
  "success": 1,
  "fail": 2
}
```

Status Code

Table 5-53 lists the status code indicating that the operation is successful. For details about the status codes indicating that the operation fails, see **Status Code**.

Table 5-53 Status code

Status Code	Description
200	The consumption of the dead letter message is successfully acknowledged.

6 FAQ

6.1 Why Is the Message "Connect IAM Timeout" Displayed When I Attempt to Access DMS?

Symptom

An error message "Connect IAM Timeout" is displayed when I attempt to use the API to access DMS.

```
Get quota fail: 401
{"message": "Connect IAM Timeout", "request_id":
"5ACB6B21-DAF6-47C8-B7A4-45A7BDC57FC6"}
```

Possible Cause

The AK/SK pair is deleted on the web-based DMS console.

Troubleshooting Method

Step 1 Log in to the management console.

Step 2 Click the username and select **My Credential** from the drop-down list.

Step 3 Click **Access Keys**.

Step 4 Click **Add Access Key**.

Step 5 Enter **Login Password** and **Verification Code** and click **OK**. Download the access key and keep it secure.

Step 6 Use the new AK/SK pair to access DMS.

----End

7 Appendix

7.1 Status Code

Table 7-1 lists status codes.

Table 7-1 Status codes

Status Code	Name	Description
100	Continue	The server has received the initial part of the request and the client should continue to send the remaining part.
101	Switching Protocols	The requester has asked the server to switch protocols and the server has agreed to do so. The target protocol must be more advanced than the source protocol. For example, the current HTTP protocol is switched to a later version of HTTP.
200	OK	Request sent successfully.
201	Created	The request has been fulfilled, resulting in the creation of a new resource.
202	Accepted	The request has been accepted for processing, but the processing has not been completed.
203	Non-Authoritative Information	The request has been fulfilled.
204	NoContent	The server has successfully processed the request, but is not returning any response body. The status code is returned in response to an HTTP OPTIONS request.

Status Code	Name	Description
205	Reset Content	The server has fulfilled the request, but the requester is required to reset the content.
206	Partial Content	The server has successfully processed a part of the GET request.
300	Multiple Choices	There are multiple options for the requested resource. For example, this code could be used to present a list of resource characteristics and addresses from which the client such as a browser may choose.
301	Moved Permanently	This and all future requests have been permanently moved to the given URI indicated in this response.
302	Found	The requested resource was temporarily moved.
303	See Other	The response to the request can be found under another URI using a GET or POST method.
304	Not Modified	The requested resource has not been modified. When the server returns this status code, it does not return any resources.
305	Use Proxy	The requested resource is available only through a proxy.
306	Unused	This HTTP status code is no longer used.
400	BadRequest	Invalid request. The client should modify the request instead of re-initiating it.
401	Unauthorized	The authorization information provided by the client is incorrect or invalid.
402	Payment Required	Reserved for future use.
403	Forbidden	The server has received the request and understood it, but the server is refusing to respond to it. The client should modify the request instead of re-initiating it.
404	NotFound	The requested resource cannot be found. The client should modify the request instead of re-initiating it.

Status Code	Name	Description
405	MethodNotAllowed	A request method is not supported for the requested resource. The client should modify the request instead of re-initiating it.
406	Not Acceptable	The server cannot fulfill the request based on the content characteristics of the request.
407	Proxy Authentication Required	This code is similar to 401, but indicates that the client must first authenticate itself with the proxy.
408	Request Time-out	The server timed out when waiting for the request. The client may re-initiate the request without any modification at any time.
409	Conflict	The request cannot be processed due to a conflict, such as an edit conflict between multiple simultaneous updates or the resource that the client attempts to create already exits.
410	Gone	The requested resource has been deleted permanently and will not be available again.
411	Length Required	The server refused to process the request because the request does not specify the length of its content.
412	Precondition Failed	The server does not meet one of the preconditions that the requester puts on the request.
413	Request Entity Too Large	The server refuses to process a request because the request is too large. The server may close the connection to prevent the client from continuing the request. If the server cannot process the request temporarily, the response will contain a Retry-After field.
414	Request-URI Too Large	The URI provided was too long for the server to process.
415	Unsupported Media Type	The server does not support the media type in the request.
416	Requested range not satisfiable	The requested range is invalid.
417	Expectation Failed	The server fails to meet the requirements of the Expect request-header field.

Status Code	Name	Description
422	UnprocessableEntity	The request is well-formed but is unable to be processed due to semantic errors.
429	TooManyRequests	The client has sent more requests than its rate limit is allowed within a given amount of time, or the server has received more requests than it is able to process within a given amount of time. In this case, the client should re-initiate requests after the time specified in the Retry-After header of the response expires.
500	InternalServerError	The server is able to receive the request but it could not understand the request.
501	Not Implemented	The server does not support the requested function.
502	Bad Gateway	The server was acting as a gateway or proxy and received an invalid request from a remote server.
503	ServiceUnavailable	The requested service is invalid. The client should modify the request instead of re-initiating it.
504	ServerTimeout	The request cannot be fulfilled within a given time. The response will reach the client only if the request carries the **timeout** parameter.
505	HTTP Version not supported	The server does not support the HTTP protocol version used in the request.

7.2 Error Code

Table 7-2 DMS error codes

Status Code	Error Code	Description
400	10240002	The number of queried queues exceeds the upper limit.
400	10240004	The tag name is invalid.
400	10240005	The project ID format is invalid.
400	10240007	The name contains invalid characters.
400	10240009	The message body is not in JSON format or contains invalid characters.
400	10240010	The description contains invalid characters.

Status Code	Error Code	Description
400	10240011	The name length must be 1 to 64 characters.
400	10240012	The name length must be 1 to 32 characters.
400	10240013	The description length must not exceed 160 characters.
400	10240014	The number of consumable messages exceeds the maximum limit.
400	10240015	The queue ID format is invalid.
400	10240016	The group ID format is invalid.
400	10240017	The queue already exists.
400	10240018	The consumer group already exists.
400	10240019	The number of consumer groups exceeds the upper limit.
400	10240020	The quota is insufficient.
400	10240021	The value of **time_wait** is not within the value range of 1–60.
400	10240022	The value of **max Consume Count** must be within the range of 1–100.
400	10240027	The value of **retention_hours** must be an integer in the range of 1–72.
400	10240028	Non-kafka queues do not support **retention_hours**.
400	10240032	The queue is being created.
401	10240101	Invalid token.
401	10240102	Expired token.
401	10240103	Missing token.
401	10240104	The project ID and token do not match.
403	10240304	Change the quota of a queue or consumer group to a value smaller than the used quota.
403	10240306	The tenant has been frozen. You cannot perform operations on DMS.
403	10240308	The queue quota must be within the range of 1–20.
403	10240309	Access denied. You cannot perform operations on DMS.
403	10240310	A tenant has read-only permissions and cannot perform operations on DMS.
403	10240311	This role does not have the permissions to perform this operation.

Status Code	Error Code	Description
403	10240312	The tenant is restricted and cannot perform operations on DMS.
404	10240401	The queue ID is incorrect or not found.
404	10240405	The consumer group ID is incorrect or not found.
404	10240406	The URL or endpoint does not exist.
500	10250002	Internal service error.
500	10250003	Internal service error.
500	10250004	Internal service error.
500	10250005	Internal communication error.
500	10250006	Internal service error.
400	10540001	The message body contains invalid fields.
400	10540003	Message ack status must be either 'success' or 'fail'. It should not be '{status}'.
400	10540004	Request error: The queue or group name does not match the handler.
400	10540010	The request format is incorrect: {error description}.
400	10540011	The message size is {message size}, larger than the size limit {max allowed size}.
400	10540012	The message body is not in JSON format or contains invalid characters.
400	10540014	The URL contains invalid parameters.
400	10540202	The request format is incorrect: {error description}.
400	10542204	Failed to consume messages due to {desc}.

Status Code	Error Code	Description
400	10542205	Failed to obtain the consumption instance because the handler does not exist. This may be because the consumer instance is released 1 minute after the message is consumed. As a result, the consumer instance fails to be obtained from the handler. This error code is returned in the following scenarios: • The handler of another consumption instance is submitted. • The message consumption is acknowledged more than 1 minute after the message is consumed. • A dead letter message consumption API is used to acknowledge the consumption of normal messages. • A normal message consumption API is used to acknowledge the consumption of dead letter messages.
400	10542206	The value of **ack_wait** must be within the range of 15–300.
400	10542209	The handler does not exist because the handler fails to be parsed, the message consumption times out, or the message consumption is repeatedly acknowledged. This error code is returned in the following scenarios: • A wrong handler is submitted after the message is consumed. • The message is not consumed and a fake handler is used to acknowledge the message consumption. • The message consumption is acknowledged more than 30s after the message is consumed. • A used handler is used to repeatedly acknowledge message consumption within 30s after the message is consumed. Due to performance concerns, the system timeout timing mechanism is not very precise, and the system performs periodic timeout detection instead of real-time detection.
400	10542214	The request format is incorrect: {error description}.
404	10240407	The request is too frequent. Flow control is being performed. Please try again later.
404	10540401	The queue name does not exist.
500	10550035	**tag_type** must be either **or** or **and**.

7.3 Obtaining a Project ID

Obtaining a Project ID by Calling an API

You can obtain a project ID by calling the API used to **query project information based on the specified criteria**.

The API for obtaining a project ID is **GET https://*{Endpoint}*/v3/projects**, where *{Endpoint}* indicates the IAM endpoint obtained from **Regions and Endpoints**. For details on API calling authentication, see **Authentication**.

The following is an example response. The value of **id** in the **projects** section is the project ID:

```
{
    "projects": [
        {
            "domain_id": "65382450e8f64ac0870cd180d14e684b",
            "is_domain": false,
            "parent_id": "65382450e8f64ac0870cd180d14e684b",
            "name": "xxx",
            "description": "",
            "links": {
                "next": null,
                "previous": null,
                "self": "https://www.example.com/v3/projects/a4a5d4098fb4474fa22cd05f897d6b99"
            },
            "id": "a4a5d4098fb4474fa22cd05f897d6b99",
            "enabled": true
        }
    ],
    "links": {
        "next": null,
        "previous": null,
        "self": "https://www.example.com/v3/projects"
    }
}
```

Obtaining a Project ID on the Console

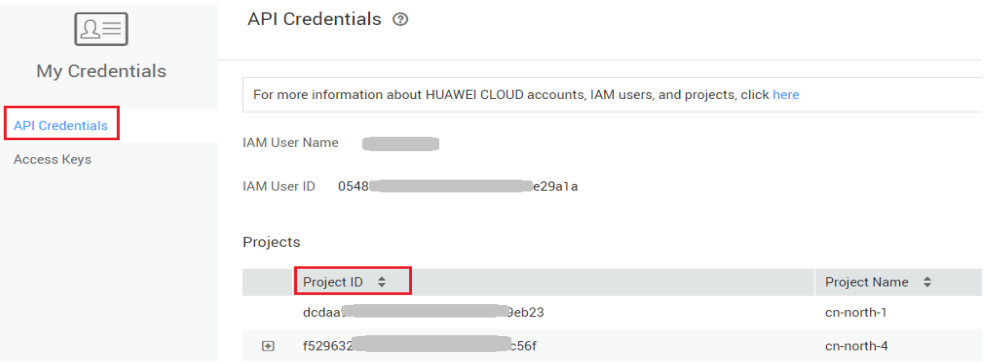
A project ID is required for some URLs when an API is called. You can obtain a project ID on the console.

The following procedure describes how to obtain a project ID:

Step 1 Log in to the management console.

Step 2 Click the username in the upper right corner, choose **My Credentials** from the drop-down list, and then view the project ID on the **Project List** tab page.

Figure 7-1 Viewing project IDs



----End

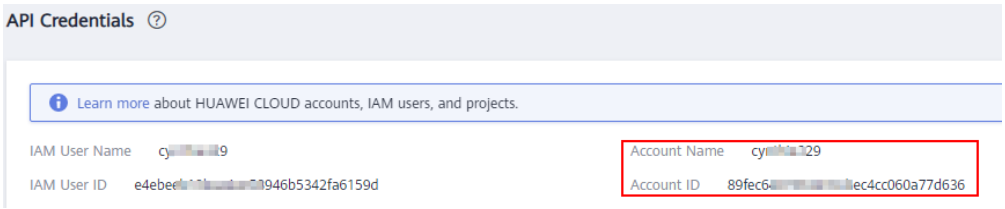
7.4 Obtaining the Domain Name and Account ID

The domain name and domain ID are required for some URLs when an API is called. To obtain the domain name and domain ID, perform the following operations:

1. Log in to the management console.
2. Click the username in the upper right corner and choose **My Credential** from the drop-down list.

   Viewing the domain name and domain ID

   Figure 7-2 Viewing the domain name and domain ID



A Change History

Released On	Description
2019-03-14	Added APIs for creating, modifying, querying, and deleting RabbitMQ instances.
2019-03-04	Added APIs for managing instances, including APIs for creating, modifying, querying, and deleting Kafka premium instances.
2019-01-04	Removed the ActiveMQ queue type.
2018-09-29	This issue is the second official release.
2018-05-10	This issue is the first official release.

