

# 消息队列 MQ

管控类 API

# 管控类 API

本文介绍 MQ Open API 的基本概念，包括使用场景、基本原理、功能以及局限性。

## 面向用户和场景

Open API 是 MQ 提供给用户的管控手段，用于实现一系列资源管理和运维功能，同生产用 SDK 不同，主要用于管理 MQ 相关的资源，例如 Topic、订阅组等。

Open API 的主要用户包括：

- 已经开通 MQ 服务，并熟悉 MQ 消息的收发。
- 已经熟悉 MQ 控制台的主要功能。
- 需要实现类似控制台的管控，存在自定义管控功能等需求。

## 适用场景和局限性

- Open API 目前仅支持公共云和金融云区域，对于私有云暂不支持，同时云上各个 Region 的开通情况请以实际公开的 Region 信息为准。
- Open API 目前属于公测阶段，稳定性和可靠性还在逐步优化，使用过程中遇到问题请联系 MQ 技术人员提供支持。
- Open API 由于其实现原理，调用端必须具备公网访问能力，否则无法调通。
- Open API 底层基于 HTTP 协议实现，性能和效率不高，仅适用于管控场景，请勿用于大批量高并发的生产场景。

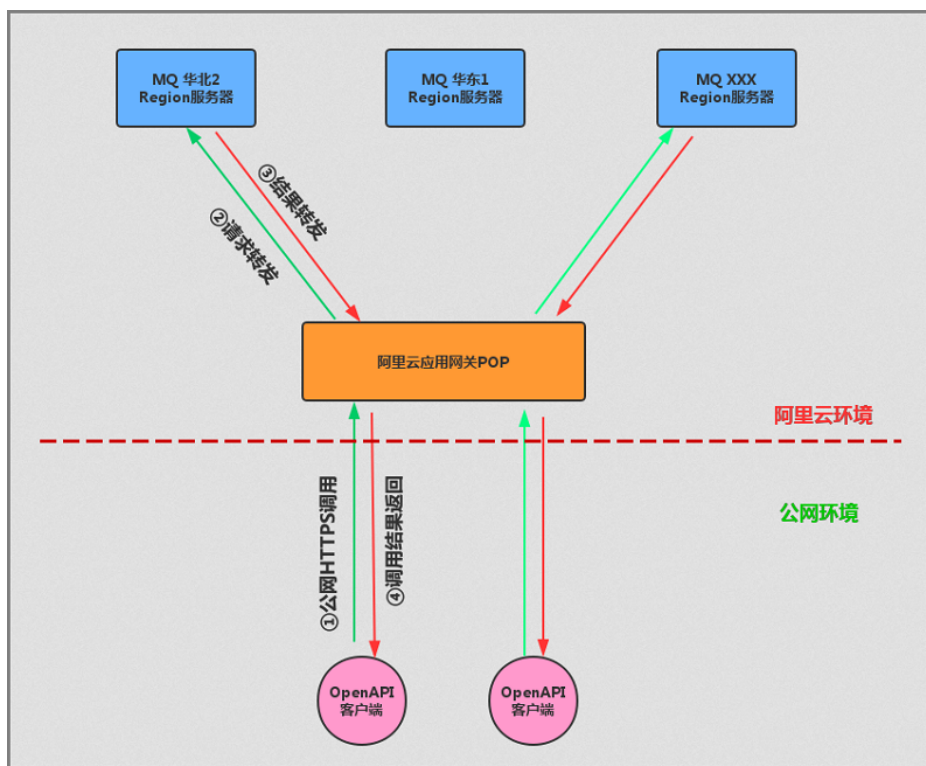
## Open API 基本原理

Open API 底层通过 HTTP 接口提供服务，用户调用 Open API 的 SDK，发出 HTTP 请求到阿里云应用网关 POP，再由 POP 将请求转发给 MQ 的后端服务去执行。

用户将参数封装到每个请求中，每个请求即对应一个接口。执行的结果放在 response 中。请求和响应的字符编码都是采用 UTF-8 编码。

**注意：**POP 网关是面对公网环境提供服务的，因此使用 Open API 的前提是客户端能够访问公网服务，否则会提示服务无法连接。

调用链路如下图所示：



#### API 版本 : 2.0.0

- 新增 消息轨迹查询接口。
- 写操作接口新增访问权限控制。

#### API 版本 : 1.3.3

- 新增 Topic 禁读禁写接口。
- 新增 ConsumerId 禁读接口。

#### API 版本 : 1.3.2-SNAPSHOT

- 修改 API 接口超时时间为最长 10 秒。
- 新增报警编辑接口。

#### API 版本 : 1.3.1-SNAPSHOT

- 支持 MQTT 相关的管控 API ;
- 所有 API 添加用户限流功能。

#### API 版本 : 1.2.1-SNAPSHOT

- 查看授权接口可以不指定 userId ;
- 支持调用账号的 parentId 查询。

#### API 版本 : 1.2.0-SNAPSHOT

- 支持 Region 化调用；
- 支持账号授权 API；
- 支持消息发送接收的趋势查询 API。

#### API 版本：1.1.1-SNAPSHOT

- 支持根据 Topic 做分页查询消息；
- 清除消息堆积支持全清和根据时间戳2种方式。

#### API 版本：1.1.0-SNAPSHOT

- Open API 支持动态错误码；
- TopicGet 接口不再需要输入 Region 信息，可以用于查询某个 Topic 属于哪个 Region；
- 该版本升级为 HTTPS 接。版本号若低于 1.1.0-SNAPSHOT，将不再支持；
- 变换版本号为 1.1.0-SNAPSHOT。

#### API 版本：1.0.2-SNAPSHOT

- Open API 修复若干 bug：
  - 创建 Topic 时，重复情况下异常信息透传；
  - 查询消息需要传入 Topic。
- 全面添加鉴权需要的信息；
- 添加消息轨迹的 API 接口；
- 报警接口，添加 list 等接口。

#### API 版本：1.0.1-SNAPSHOT

- Open API 修复若干 bug：
  - 把返回结果中所有 regionId 替换成 OnsRegionId 以和 POP 的 regionId 区分开，方便用户理解；
  - 修复 consumerstatus 返回结果中的若干问题，保护 consumetime 超界，clientId 没有说明等问题；
  - 修复 consumestatus 等接口返回的 List 为空等问题。
- 变换版本号为 1.0.1-SNAPSHOT

#### API 版本：1.0.0-SNAPSHOT

- Open API 上线测试；
- 更新文档中的参数说明，修复 regionidList 等方法的使用；
- 添加接入 API 的依赖描述。

本文档介绍 MQ Open API 的接入步骤以及注意事项，包含 SDK 的获取，以及初始化参数的设置。

## 1. SDK 获取

直接填写以下 POM 的配置，依赖 Open API 的 SDK 即可。

```
<dependencies>
<dependency>
<groupId>com.aliyun</groupId>
<artifactId>aliyun-java-sdk-core</artifactId>
<optional>true</optional>
<version>3.2.8</version>
</dependency>
<dependency>
<groupId>com.aliyun</groupId>
<artifactId>aliyun-java-sdk-ons</artifactId>
<version>2.0.0</version>
</dependency>
</dependencies>
```

## 2. 公共参数设置

Open API 使用时需要启动 Open API 的客户端，而客户端启动时需要设置接入点和 AccessKey，SecretKey 等参数信息，具体示例如下：

```
/**
 *Open API的接入点，设置为目标Region
 */
String regionId = "XXXXX";
/**
 *鉴权使用的AccessKey，由阿里云官网控制台获取
 */
String accessKey = "XXXXXXXXXXXXXXXXXXXX";
/**
 *鉴权使用的SecretKey，由阿里云官网控制台获取
 */
String secretKey = "XXXXXXXXXXXXXXXXXXXX";
/*
 *通过Open API访问的云产品名称，此处设置为Ons即可
 */
String productName ="Ons";
/**
 *对应endPoint接入点的接入点域名
 */
String domain ="ons.XXXXX.aliyuncs.com";
try {
DefaultProfile.addEndpoint(regionId , regionId , productName , domain);
} catch (ClientException e) {
e.printStackTrace();
}
IClientProfile profile= DefaultProfile.getProfile(regionId , accessKey , secretKey);
IAcsClient iAcsClient= new DefaultAcsClient(profile);
```

### 参数说明

- regionId：指的是 API 的网关所在区域，目前支持的有 cn-beijing，cn-hangzhou，cn-qingdao，cn-shenzhen 等。
- accessKey：指的是用户在阿里云服务器管理控制台上获取的 AK。

- secretKey：指的是用户在阿里云服务器管理控制台上获得的 SK。
- endPointName：接入点名称，同 RegionId 一致即可。
- productName：Open API 的产品名称，直接写 Ons 即可。
- domain：Open API 的接入点 Domain，规则是ons.\${RegionId}.aliyuncs.com，目前 MQ 支持 Open API 的 Region 请参考下表。

| 区域名称   | RegionId            | Domain                               | 备注                        |
|--------|---------------------|--------------------------------------|---------------------------|
| 公共云华北2 | cn-beijing          | ons.cn-beijing.aliyuncs.com          | 使用公共云华北2 Region 的用户建议此接入点 |
| 公网     | cn-qingdao          | ons.cn-qingdao.aliyuncs.com          | 使用公网 Region 的用户建议此接入点     |
| 公共云华东1 | cn-hangzhou         | ons.cn-hangzhou.aliyuncs.com         | 使用公共云华东1 Region 的用户建议此接入点 |
| 公共云华东2 | cn-shanghai         | ons.cn-shanghai.aliyuncs.com         | 使用公共云华东2 Region 的用户建议此接入点 |
| 公共云华南1 | cn-shenzhen         | ons.cn-shenzhen.aliyuncs.com         | 使用公共云华南1 Region 的用户建议此接入点 |
| 公共云新加坡 | ap-southeast-1      | ons.ap-southeast-1.aliyuncs.com      | 使用新加坡 Region 的用户建议此接入点    |
| 金融云华东1 | cn-hangzhou-finance | ons.cn-hangzhou-finance.aliyuncs.com | 使用金融云华东1 Region 的用户建议此接入点 |
| 金融云华南1 | cn-shenzhen-finance | ons.cn-shenzhen-finance.aliyuncs.com | 使用金融云华南1 Region 的用户建议此接入点 |
| 金融云华东2 | cn-shanghai-finance | ons.cn-shanghai-finance.aliyuncs.com | 使用金融云华东2 Region 的用户建议此接入点 |

本文档描述 MQ Open API 涉及到的公共术语的解释，供查询参考。

## 网关参数类

### RegionId

阿里云上产品所在的地域，包括 cn-beijing，cn-hangzhou 等多个，调用 Open API 时请根据访问需求设置对应的 Region，例如需要操作华北2 Region 的 MQ 资源请设置 cn-beijing。相关信息见首页的表格。

### OnsRegionId

MQ 产品定义的 Region 信息，一般用于 API 请求的参数，用于指定 API 操作对应 Region 的 MQ 资源，此处 Region 信息需要根据 OnsRegionList 接口获取。

### EndPoint

Open API 根据设置的接入点来调用对应的后端服务。一般设置为对应的 RegionId 即可。

### Domain

Open API 根据设置的访问域名来调用对应的后端服务。例如 domain 为 ons.cn-hangzhou.aliyuncs.com 代表连到华东1的 MQ 后端服务。具体域名信息参考接入指南。

## 账号安全类

### AccessKey

访问阿里云产品使用的账号的 AK，由阿里云官网获得，调用时传入给 MQ 后台用于身份验证，调用 MQ Open API 只能操作对应账号下的资源。

### SecretKey

访问阿里云产品使用的账号的 AK 密钥，由阿里云官网获得，调用时传入，用于计算访问请求的签名，MQ 后端根据对应账号比对签名实现鉴权。

## MQ 资源管控类

### Topic

MQ 产品中消息按照 Topic 进行分来存储，因此 Topic 是 MQ 中最重要的资源之一。MQ 中 Topic 全局唯一。

### ProducerId(PID)

MQ 产品中确定发布关系的唯一标识，用于标识一个发送的集群。MQ 中 ProducerId 全局唯一，后续文档中对 ProducerID 简称 PID。

### ConsumerId(CID)

MQ 产品中确定订阅关系的唯一标识，用于标识一个订阅组，而采用同一个 ConsumerId 的客户端，其消费消息的逻辑必须完全一致。MQ 中 ConsumerId 全局唯一，后续文档中对 ConsumerId 简称 CID。

### 消息堆积

消息堆积指 MQ 产品中描述消费方尚未进行处理的消息数据，当消息的生产快于消费的时候，两者之差即为堆积数，正常情况下堆积数应该在可控范围内。如果堆积数过多而且成上涨趋势，则需要关注消费方慢的原因。

## 场景示例

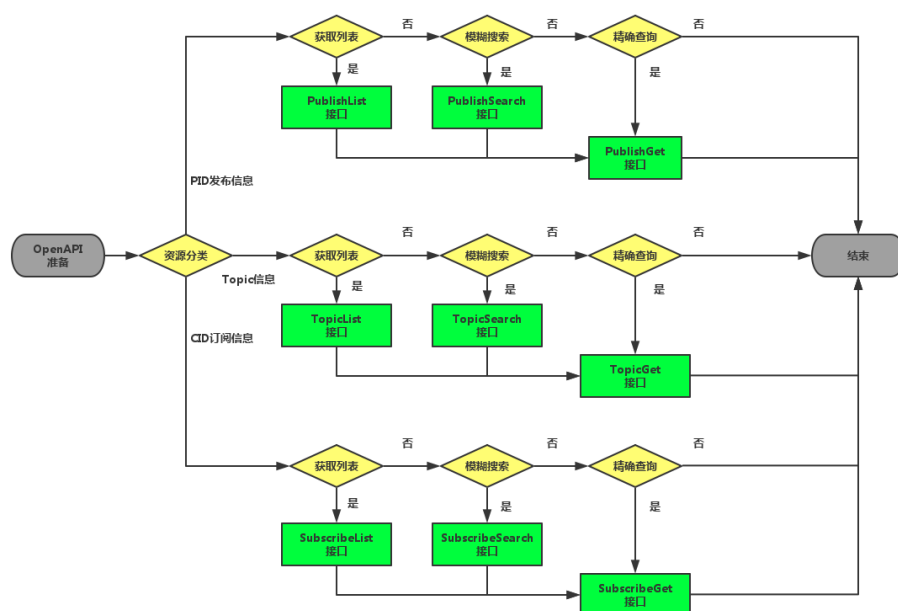
本文档主要介绍如何使用 Open API 来完成对指定账号下所拥有的 MQ 资源的查询，包含三类，分别是 Topic，Publish，Subscribe 信息。

## 资源查询

1. 查询的资源包含 Topic，Publish，Subscribe 三大类信息。
2. 查询时均可根据 Topic，Publish，Subscribe 的名称进行精确查询。
3. 如果没有精确信息，可以根据相关内容自动进行搜索查询。
4. 如果不需要获取精确信息，可以查询当前账号下所有资源的列表。

## 主要流程

调用流程如下图所示：



## 相关 API

- Publish 列表: OnsPublishList
- Publish 搜索: OnsPublishSearch
- Publish 精确查询: OnsPublishGet
- Topic 列表: OnsTopicList
- Topic 搜索: OnsTopicSearch
- Topic 精确查询: OnsTopicGet
- Subscribe 列表: OnsSubscribeList
- Subscribe 搜索: OnsSubscribeSearch
- Subscribe 精确查询: OnsSubscribeGet

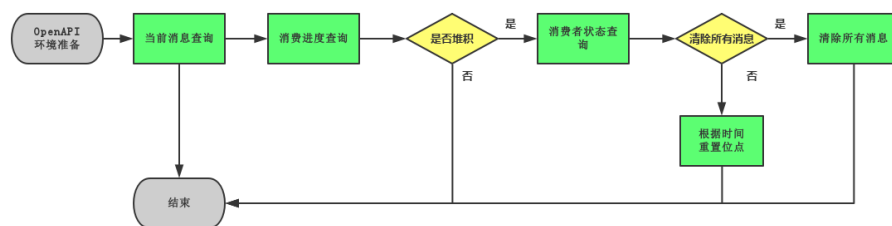
本文主要介绍如何使用 Open API 来完成运行过程中 MQ 消息状态的查询，一般用于定位消息发送状况、消息消费状况，是否有堆积，如何处理堆积等。

## 查询思路

1. 查看 Topic 消息发送状态：即查看目标 Topic 当前服务器上存在的有效消息总数以及最新状态。
2. 查看订阅组的消费进度：即查看指定的订阅组的消费是否有堆积。
3. 客户端消费状态查询：如果存在堆积，可以查看消费组内各个客户端的连接状态，是否有客户端消费线程阻塞。
4. 消费位点重置：根据业务需求对消费方的消费进度位点进行更新，该功能包含两个场景。
  - 场景 A：如果消息可以丢弃，则选择清除所有消息。
  - 场景 B：如果堆积的消息需要回收或者更新到指定时刻，可以选择根据指定时刻进行位点重置。

## 主要流程

调用流程如下图所示：



## 相关 API

- 当前消息查询: OnsTopicStatus
- 消费进度查询: OnsConsumerAccumulate
- 消费者状态查询: OnsConsumerStatus
- 清除堆积: OnsConsumerResetOffset

本文档主要介绍如何使用 Open API 来完成指定 Topic 资源的授权操作。

## 资源授权

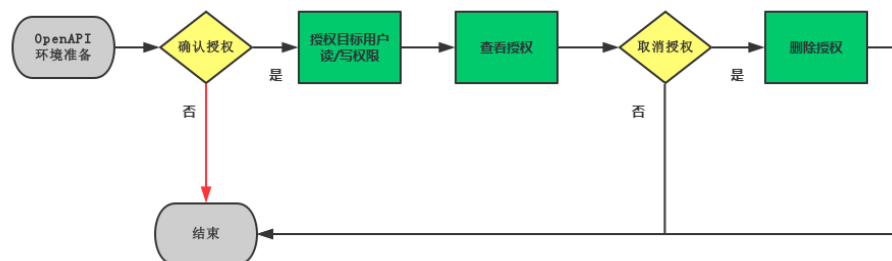
**新建授权：**将自己当前所拥有的指定 Topic 授权给目标用户，可以是主账号或者子账户。

**查看授权：**查看目标 Topic 已经被授权的用户列表以及权限数据。

**删除授权：**删除某个 Topic 对目标账号的授权。

## 主要流程

调用流程如下图所示：



## 相关 API

- 新建授权: OnsEmpowerCreate
- 查看授权: OnsEmpowerList
- 删除授权: OnsEmpowerDelete

MQ Open API 调用成功时返回给客户端的 ResponseCode=200，当出现调用失败时，会返回对应的异常错误码以及描述信息。您可以根据以下异常信息对照表查找对应的解决方法。

| 错误码                        | 现象&原因                 | 解决方法                     |
|----------------------------|-----------------------|--------------------------|
| ONS_SYSTEM_ERROR           | MQ 后端异常               | 通过阿里云工单联系 MQ 技术人员        |
| ONS_SERVICE_UN SUPPORTED   | 当前调用在对应的 Region 区域不支持 | 请找 MQ 技术支持确认该接口的开通情况     |
| ONS_INVOKE_ERROR           | Open API 接口调用失败       | 联系 MQ 技术人员处理             |
| BIZ_FIELD_CHECK_INVALID    | 参数检验失败                | 请参考 API 手册检查各个参数传入是否合法   |
| BIZ_TOPIC_NOT_FOUND        | Topic 没有找到            | 请检查 Topic 输入是否合法，或者是否创建过 |
| BIZ_SUBSCRIPTION_NOT_FOUND | 目标订阅关系 CID 找不到        | 请检查 CID 是否创建过，或者查询条件错误   |
| BIZ_PUBLISHER_EXISTED      | 指定 PID 已经存在           | 更换 PID 的名称重试请求           |
| BIZ_SUBSCRIPTION_EXISTED   | 指定 CID 已经存在           | 更换 CID 的名称重试请求           |
| BIZ_CONSUMER_NOT_ONLINE    | 指定 CID 的客户端不在线        | 请确保消费端在线，然后重试请求          |
| BIZ_NO_MESSAGE             | 当前查询条件没有匹配消息          | 请检查查询条件，并确认查询范围内是否发过消息   |

|                                      |                        |                      |
|--------------------------------------|------------------------|----------------------|
| BIZ_REGION_NOT_FOUND                 | 请求的 Region 找不到         | 确认请求的 Region 参数是否合法  |
| BIZ_TOPIC_EXISTED                    | 指定 Topic 已经存在          | 更改 Topic 名称重试请求      |
| BIZ_PRODUCER_ID_BELONG_TO_OTHER_USER | 当前 PID 已经被其他用户占用       | 更换 PID 重试请求          |
| BIZ_CONSUMER_ID_BELONG_TO_OTHER_USER | 当前 CID 已经被其他用户占用       | 更换CID重试请求            |
| BIZ_PUBLISH_INFO_NOT_FOUND           | 请求的 PID 没有找到           | 确认 PID 是否存在，或者请求条件错误 |
| EMPOWER_EXIST_ERROR                  | 当前授权关系已经存在             | 核实请求参数，重试或者先查询       |
| EMPOWER_OWNER_CHECK_ERROR            | 当前用户不是授权 Topic 的 Owner | 确认资源所属关系             |

## 用户管理接口

OnsRegionList 接口用于返回 MQ 目前开放服务的 Region 列表。

### 使用场景

在所有 Open API 的调用过程中都需要指定 MQ 的 Region 信息。该 Region 字段即可通过 OnsRegionList 接口获取。

### 请求参数列表

| 名称           | 类型     | 是否必须 | 描述                     |
|--------------|--------|------|------------------------|
| OnsRegionId  | String | 否    | 当前查询 MQ 所在区域           |
| OnsPlatform  | String | 否    | 该请求来源，默认是从 POP 平台      |
| PreventCache | Long   | 是    | 用于 CSRF 校验，设置为系统当前时间即可 |

### 返回参数列表

| 名称        | 类型             | 描述             |
|-----------|----------------|----------------|
| RequestId | String         | 为公共参数，每个请求独一无二 |
| HelpUrl   | String         | 帮助链接           |
| Data      | List(RegionDo) | 查询结果           |

## RegionDo 数据结构

| 成员          | 类型     | 描述            |
|-------------|--------|---------------|
| Id          | Long   | 数据库中编号        |
| RegionId    | String | regionId 名称   |
| RegionName  | String | Region 别名     |
| ChannelId   | Long   | 所在 Channel 编号 |
| ChannelName | String | 所在 Channel 别名 |
| CreateTime  | Long   | 创建时间          |
| UpdateTime  | Long   | 最后更新时间        |

## 使用示例

```
public static void main(String[] args) {
    String regionId = "cn-hangzhou";
    String accessKey = "XXXXXXXXXXXXXXXXXX";
    String secretKey = "XXXXXXXXXXXXXXXXXX";
    String endPointName = "cn-hangzhou";
    String productName = "Ons";
    String domain = "ons.cn-hangzhou.aliyuncs.com";

    /**
     *根据自己需要访问的区域选择Region，并设置对应的接入点
     */
    try {
        DefaultProfile.addEndpoint(endPointName, regionId, productName, domain);
    } catch (ClientException e) {
        e.printStackTrace();
    }
    IClientProfile profile = DefaultProfile.getProfile(regionId, accessKey, secretKey);
    IAcsClient IAcsClient = new DefaultAcsClient(profile);
    OnsRegionListRequest request = new OnsRegionListRequest();
    request.setAcceptFormat(FormatType.JSON);
    request.setPreventCache(System.currentTimeMillis());
    try {
        OnsRegionListResponse response = IAcsClient.getAcsResponse(request);
        List<OnsRegionListResponse.RegionDo> regionDoList = response.getData();
        for (OnsRegionListResponse.RegionDo regionDo : regionDoList) {
            System.out.println(regionDo.getId() + " " +
                regionDo.getOnsRegionId() + " " +
                regionDo.getRegionName() + " " +
                regionDo.getChannelId() + " " +
                regionDo.getChannelName() + " " +
                regionDo.getCreateTime() + " " +
                regionDo.getUpdateTime());
        }
    } catch (ClientException e) {
        e.printStackTrace();
    }
}
```

```
}
catch (Exception e) {
e.printStackTrace();
}
}
```

# Topic 管理接口

本接口限企业铂金版客户专用，请前往铂金版购买页面查看详情。

## 描述

Topic 创建接口一般是在有新的 Topic 需求的情况下调用，例如发布新应用，业务逻辑需要新的 Topic。  
Topic 为全局唯一，不可重复创建，否则会创建失败，需要根据错误码进行处理。

## 主子帐号约束

本接口默认仅限主账号使用，RAM 子账号无法使用，除非该 RAM 子账号被授予了相关权限。

## 请求参数

| 名称           | 类型      | 是否必须 | 描述                               |
|--------------|---------|------|----------------------------------|
| OnsRegionId  | String  | 是    | 当前查询 MQ 所在区域，详情请参考公共术语说明         |
| OnsPlatform  | String  | 否    | 该请求来源，默认是从 POP 平台                |
| PreventCache | Long    | 是    | 用于 CSRF 校验，设置为系统当前时间即可           |
| Topic        | String  | 是    | 需要创建的 Topic 名称，Topic 名称全局唯一，不可重复 |
| Qps          | Long    | 否    | 设置该 Topic 的 QPS 估计               |
| Status       | Integer | 否    | Topic 申请的状态, ( 0 服务中 1 冻结 2 暂停 ) |
| Remark       | String  | 否    | 备注，可以不填                          |
| Appkey       | String  | 否    | 应用的 Key，公共云不填，聚石塔用户需要设置          |

|         |        |   |           |
|---------|--------|---|-----------|
| AppName | String | 否 | 应用名，公共云不填 |
|---------|--------|---|-----------|

## 返回参数

| 名称        | 类型     | 描述                      |
|-----------|--------|-------------------------|
| RequestId | String | 为公共参数，每个请求独一无二，用于排查定位问题 |
| HelpUrl   | String | 帮助链接                    |

## 相关 API

- OnsTopicDelete：删除 Topic
- OnsTopicList：查看 Topic 列表
- OnsPublishCreate：在创建 Topic 之后一般需要创建发布关系
- OnsSubscribeCreate：在创建 Topic 之后一般需要创建订阅关系

## 使用示例

本示例仅提供一个参考，从华东1接入点接入，创建名为“XXX”的 Topic，创建到 daily 区域。

```
public static void main(String[] args) {
    String regionId = "cn-hangzhou";
    String accessKey = "XXXXXXXXXXXXXXXXXX";
    String secretKey = "XXXXXXXXXXXXXXXXXX";
    String endPointName = "cn-hangzhou";
    String productName = "Ons";
    String domain = "ons.cn-hangzhou.aliyuncs.com";

    /**
     *根据自己需要访问的区域选择Region，并设置对应的接入点
     */
    try {
        DefaultProfile.addEndpoint(endPointName, regionId, productName, domain);
    } catch (ClientException e) {
        e.printStackTrace();
    }
    IClientProfile profile = DefaultProfile.getProfile(regionId, accessKey, secretKey);
    IAcsClient iAcsClient = new DefaultAcsClient(profile);

    OnsTopicCreateRequest request = new OnsTopicCreateRequest();
    request.setAcceptFormat(FormatType.JSON);
    request.setTopic("XXXXXXXXXX");
    request.setQps(1000l);
    request.setRemark("test");
    request.setStatus(0);
    /**
     *ONSRegionId是指需要API访问MQ哪个区域的资源
     *该值必须要根据OnsRegionList方法获取的列表来选择和配置，因为OnsRegionId是变动的，不能够写固定值
     */
    request.setOnsRegionId("daily");
}
```

```
request.setPreventCache(System.currentTimeMillis());
try {
    OnsTopicCreateResponse response = client.getAcsResponse(request);
    System.out.println(response.getRequestId());
}
catch (ServerException e) {
    e.printStackTrace();
}
catch (ClientException e) {
    e.printStackTrace();
}
catch (Exception e) {
    e.printStackTrace();
}
}
```

本接口限企业铂金版客户专用，请前往铂金版购买页面查看详情。

## 描述

OnsTopicDelete 接口用于删除账号下的指定 Topic。

Topic 删除接口一般是资源回收时使用，例如应用下线。Topic 删除时 MQ 后台需要做资源回收，速度较慢，因此不建议删除后立即又重新创建。如果删除失败，请根据错误码进行处理。

**注意：**删除 Topic 将导致该 Topic 下所有的发布订阅关系同时被清理，请慎重调用。

## 主子帐号约束

本接口默认仅限主账号使用，RAM 子账号无法使用，除非该 RAM 子账号被授予了相关权限。

## 请求参数

| 名称           | 类型     | 是否必须 | 描述                                             |
|--------------|--------|------|------------------------------------------------|
| OnsRegionId  | String | 是    | 当前查询 MQ 所在区域，可以通过 OnsRegionList 方法获取，请参考公共术语说明 |
| OnsPlatform  | String | 否    | 该请求来源，默认是从 POP 平台                              |
| PreventCache | Long   | 是    | 用于 CSRF 校验，设置为系统当前时间即可                         |
| Topic        | String | 是    | 指定删除的 Topic                                    |

## 返回参数

| 名称 | 类型 | 描述 |
|----|----|----|
|----|----|----|

|           |        |                         |
|-----------|--------|-------------------------|
| RequestId | String | 为公共参数，每个请求独一无二，用于排查定位问题 |
| HelpUrl   | String | 帮助链接                    |

## 相关 API

- OnsTopicCreate：创建 Topic
- OnsTopicList：查看 Topic 列表
- OnsPublishDelete：删除发布关系
- OnsSubscribeDelete：删除订阅关系

## 使用示例

本示例仅提供一个参考，从华东1接入点接入，删除名为“Mingduan\_67dd”的 Topic。

### 调用示例

```
public static void main(String[] args){
String regionId = "cn-hangzhou";
String accessKey = "XXXXXXXXXXXXXXXXXXXX";
String secretKey = "XXXXXXXXXXXXXXXXXXXX";
String endPointName = "cn-hangzhou";
String productName = "Ons";
String domain = "ons.cn-hangzhou.aliyuncs.com";

/**
*根据自己需要访问的区域选择Region，并设置对应的接入点
*/
try {
DefaultProfile.addEndpoint(endPointName, regionId, productName, domain);
} catch (ClientException e) {
e.printStackTrace();
}
IClientProfile profile= DefaultProfile.getProfile(regionId, accessKey, secretKey);
IAcsClient iAcsClient= new DefaultAcsClient(profile);
OnsTopicDeleteRequest request =new OnsTopicDeleteRequest();
request.setCluster("taobaodaily");
request.setPreventCache(System.currentTimeMillis());
/**
*ONSRegionId是指设置API访问MQ特点的区域
*该值必须要根据OnsRegionList方法获取的列表来选择和配置，因为OnsRegionId是变动的，不能够写固定值
*/
request.setOnsRegionId("daily");
request.setTopic("Mingduan_67dd");
try {
OnsTopicDeleteResponse response = client.getAcsResponse(request);
System.out.println(response.getRequestId());
}
catch (ServerException e) {
e.printStackTrace();
}
catch (ClientException e) {
```

```
e.printStackTrace();
}
catch (Exception e) {
e.printStackTrace();
}
}
```

OnsTopicList 接口用于查询当前用户下所有 Topic 的列表信息。

## 使用场景

查询账号下所有 Topic 的列表信息，一般用于生成资源列表的场景，并不查看具体信息。

## 请求参数列表

| 名称           | 类型     | 是否必须 | 描述                                          |
|--------------|--------|------|---------------------------------------------|
| OnsRegionId  | String | 是    | 当前查询 MQ 所在区域，可以通过 <b>ONSRegionList</b> 方法获取 |
| OnsPlatform  | String | 否    | 该请求来源，默认是从 POP 平台                           |
| PreventCache | Long   | 是    | 用于 CSRF 校验，设置为系统当前时间即可                      |
| Topic        | String | 否    | 查询指定 Topic 时设置，否则查询所有 Topic                 |

## 返回参数列表

| 名称        | 类型                  | 描述                  |
|-----------|---------------------|---------------------|
| RequestId | String              | 为公共参数，每个请求独一无二      |
| HelpUrl   | String              | 帮助链接                |
| Data      | List(PublishInfoDo) | 返回所有已发布的 Topic 信息列表 |

## PublishInfoDo 数据结构

| 成员          | 类型      | 描述                                          |
|-------------|---------|---------------------------------------------|
| id          | Long    | 该 Topic 信息编号                                |
| channelId   | Integer | 该 Topic 所在区域 ID，0代表阿里云，1代表聚石塔               |
| channelName | String  | 该 Topic 所在区域名称，ALIYUN 代表阿里云，CLOUD 代表聚石塔，... |

|              |         |                                          |
|--------------|---------|------------------------------------------|
| onsRegionId  | Long    | 该 Topic 所在区域 ID，就是 ONSRegionList 方法获取的内容 |
| regionName   | String  | 该 Topic 所在区域名称                           |
| topic        | String  | Topic 名称                                 |
| owner        | String  | Topic 所有者编号，为阿里云的 uid                    |
| relation     | Integer | 所有关系编号1为持有者，2表示可以发布，4表示可以订阅，6表示可以发布和订阅   |
| relationName | String  | 所有关系名称，例如持有者、可订阅、可发布、可发布订阅               |
| status       | Integer | 当前状态编号（0 服务中 1 冻结 2 暂停）                  |
| statusName   | String  | 当前状态别名，服务中，                              |
| appkey       | String  | null                                     |
| createTime   | Long    | 创建时间                                     |
| updateTime   | Long    | 更新时间                                     |
| remark       | String  | 备注信息                                     |

## 相关 API

- OnsTopicCreate：创建 Topic
- OnsTopicList：查看 Topic 列表
- OnsPublishDelete：删除发布关系
- OnsSubscribeDelete：删除订阅关系

## 使用示例

本示例仅提供一个参考，从华东1接入点接入，查询当前用户下所有的 Topic 列表。

```
public static void main (String[]args) {
    String regionId = "cn-hangzhou";
    String accessKey = "XXXXXXXXXXXXXXXXXX";
    String secretKey = "XXXXXXXXXXXXXXXXXX";
    String endPointName = "cn-hangzhou";
    String productName = "Ons";
    String domain = "ons.cn-hangzhou.aliyuncs.com";

    /**
     *根据自己需要访问的区域选择Region，并设置对应的接入点
     */
    try {
        DefaultProfile.addEndpoint(endPointName, regionId, productName, domain);
```

```
    } catch (ClientException e) {
    e.printStackTrace();
    }
    IClientProfile profile = DefaultProfile.getProfile(regionId , accessKey , secretKey);
    IAcsClient IAcsClient = new DefaultAcsClient(profile);
    OnsTopicListRequest request = new OnsTopicListRequest();
    /**
    *ONSRegionId是指你需要API访问MQ哪个区域的资源.
    *该值必须要根据OnsRegionList方法获取的列表来选择和配置，因为OnsRegionId是变动的，不能够写固定值
    */
    request.setOnsRegionId("daily");
    request.setPreventCache(System.currentTimeMillis());
    request.setTopic("XXXXXXXXXXXXXX");
    try {
    OnsTopicListResponse response = IAcsClient.getAcsResponse(request);
    List<OnsTopicListResponse.PublishInfoDo> publishInfoDoList=response.getData();
    for(OnsTopicListResponse.PublishInfoDo publishInfoDo:publishInfoDoList){
    System.out.println(publishInfoDo.getTopic()+" "+ publishInfoDo.getOwner());
    }
    } catch (ServerException e) {
    e.printStackTrace();
    } catch (ClientException e) {
    e.printStackTrace();
    }
    }
```

## 查询 Topic

OnsTopicGet 接口用于查询指定 Topic 的信息，属于精确查询。

### 使用场景

查询账号下指定 Topic 的分布情况，并获取 Topic 当前的开通状态和权限关系等信息。由于是精确查询，因此输入参数必须包含 Topic 名称。

### 请求参数列表

| 名称           | 类型     | 是否必须 | 描述                                         |
|--------------|--------|------|--------------------------------------------|
| OnsRegionId  | String | 是    | 当前查询 MQ 所在区域，可以通过ONSRegionList 方法获取，参见公共术语 |
| OnsPlatform  | String | 否    | 该请求来源，默认是从POP 平台                           |
| PreventCache | Long   | 是    | 用于 CSRF 校验，设置为系统当前时间即可                     |
| Topic        | String | 是    | 指定查询的 Topic                                |

## 返回参数列表

| 名称        | 类型                  | 描述                  |
|-----------|---------------------|---------------------|
| RequestId | String              | 为公共参数，每个请求独一无二      |
| HelpUrl   | String              | 帮助链接                |
| Data      | List(PublishInfoDo) | 返回所有已发布的 Topic 信息列表 |

## PublishInfoDo 数据结构

| 成员           | 类型      | 描述                                       |
|--------------|---------|------------------------------------------|
| id           | Long    | 该 Topic 信息编号                             |
| channelId    | Integer | 该 Topic 所在区域 ID，0代表阿里云，1代表聚石塔            |
| channelName  | String  | 该 Topic 所在类别，阿里云，或者聚石塔，...               |
| onsRegionId  | Long    | 该 Topic 所在区域 ID，就是 ONSRegionList 方法获取的内容 |
| regionName   | String  | 该 Topic 所在区域名称                           |
| topic        | String  | Topic 名称                                 |
| owner        | String  | Topic 所有者编号，为阿里云的 uid                    |
| relation     | Integer | 所有关系编号1为持有者，2为可以发布，4为可以订阅，6为可以发布和订阅      |
| relationName | String  | 所有关系名称，例如持有者、可订阅、可发布、可发布订阅               |
| status       | Integer | 当前状态编号（0 服务中 1 冻结 2 暂停）                  |
| statusName   | String  | 当前状态别名，服务中，未开通                           |
| appkey       | String  | null                                     |
| createTime   | Long    | 创建时间                                     |
| updateTime   | Long    | 更新时间                                     |
| remark       | String  | 备注信息                                     |

## 相关 API

- OnsTopicCreate：创建 Topic
- OnsTopicList：查看 Topic 列表

- OnsPublishDelete：删除发布关系
- OnsSubscribeDelete：删除订阅关系

## 使用示例

本示例仅提供一个参考，从华东1接入点接入，查询名为“MingduanTest”的Topic，并打印Topic的相关信息。

```
public static void main(String[] args) {
    String regionId = "cn-hangzhou";
    String accessKey = "XXXXXXXXXXXXXXXXXX";
    String secretKey = "XXXXXXXXXXXXXXXXXX";
    String endPointName = "cn-hangzhou";
    String productName = "Ons";
    String domain = "ons.cn-hangzhou.aliyuncs.com";

    /**
     *根据自己需要访问的区域选择Region，并设置对应的接入点
     */
    try {
        DefaultProfile.addEndpoint(endPointName, regionId, productName, domain);
    } catch (ClientException e) {
        e.printStackTrace();
    }
    IClientProfile profile = DefaultProfile.getProfile(regionId, accessKey, secretKey);
    IAcsClient IAcsClient = new DefaultAcsClient(profile);

    OnsTopicGetRequest request = new OnsTopicGetRequest();
    request.setAcceptFormat(FormatType.JSON);
    request.setTopic("MingduanTest");
    /**
     *ONSRegionId是指需要API访问MQ哪个区域的资源.
     *该值必须要根据OnsRegionList方法获取的列表来选择和配置，因为OnsRegionId是变动的，不能够写固定值
     */
    request.setOnsRegionId("daily");
    request.setPreventCache(System.currentTimeMillis());
    try {
        OnsTopicGetResponse response = IAcsClient.getAcsResponse(request);
        List<OnsTopicGetResponse.PublishInfoDo> publishInfoDoList = response.getData();
        for (OnsTopicGetResponse.PublishInfoDo publishInfoDo : publishInfoDoList) {
            System.out.println(publishInfoDo.getId() + " " +
                publishInfoDo.getChannelId() + " " +
                publishInfoDo.getChannelName() + " " +
                publishInfoDo.getOnsRegionId() + " " +
                publishInfoDo.getRegionName() + " " +
                publishInfoDo.getTopic() + " " +
                publishInfoDo.getOwner() + " " +
                publishInfoDo.getRelation() + " " +
                publishInfoDo.getRelationName() + " " +
                publishInfoDo.getStatus() + " " +
                publishInfoDo.getStatusName() + " " +
                publishInfoDo.getAppkey() + " " +
                publishInfoDo.getCreateTime() + " " +
```

```

publishInfoDo.getUpdateTime()+" "+
publishInfoDo.getRemark());
}
System.out.println(response.getRequestId());
}
catch (ServerException e) {
e.printStackTrace();
}
catch (ClientException e) {
e.printStackTrace();
}
catch (Exception e) {
e.printStackTrace();
}
}
}

```

OnsTopicSearch 接口用于根据搜索条件搜索指定账号下所有的 Topic 信息列表。

## 使用场景

该接口适用于查询 Topic 信息时没有精确的 Topic 名称的情况，一般用于获取整个 Topic 列表，在选择特定的 Topic 后基于 OnsTopicGet 接口精确查询。

## 请求参数列表

| 名称           | 类型     | 是否必须 | 描述                                          |
|--------------|--------|------|---------------------------------------------|
| OnsRegionId  | String | 是    | 当前查询 MQ 所在区域，可以通过 ONSRegionList 方法获取，参见公共术语 |
| OnsPlatform  | String | 否    | 该请求来源，默认是从 POP 平台                           |
| PreventCache | Long   | 是    | 用于 CSRF 校验，设置为系统当前时间即可                      |
| Search       | String | 是    | 输入的搜索条件，可以是 Topic 或者 appname 等字段            |

## 3. 返回参数列表

| 名称        | 类型                  | 描述                  |
|-----------|---------------------|---------------------|
| RequestId | String              | 为公共参数，每个请求独一无二      |
| HelpUrl   | String              | 帮助链接                |
| Data      | List(PublishInfoDo) | 返回所有已发布的 Topic 信息列表 |

PublishInfoDo 数据结构

| 成员           | 类型      | 描述                                      |
|--------------|---------|-----------------------------------------|
| id           | Long    | 该 Topic 信息编号                            |
| channelId    | Integer | 该 Topic 所在区域 ID，0代表阿里云，1代表聚石塔           |
| channelName  | String  | 该 Topic 所在区域名称，ALIYUN，CLOUD，...         |
| onsRegionId  | Long    | 该 Topic 所在区域 ID，就是ONSRegionList 方法获取的内容 |
| regionName   | String  | 该 Topic 所在区域名称                          |
| topic        | String  | Topic 名称                                |
| owner        | String  | Topic 所有者编号，为阿里云的 uid                   |
| relation     | Integer | 所有关系编号1为持有者，2为可以发布，4为可以订阅，6为可以发布和订阅     |
| relationName | String  | 所有关系名称，例如持有者、可订阅、可发布、可发布订阅              |
| status       | Integer | 当前状态编号（0 服务中 1 冻结 2 暂停）                 |
| statusName   | String  | 当前状态别名，服务中，                             |
| appkey       | Integer | null                                    |
| createTime   | Long    | 创建时间                                    |
| updateTime   | Long    | 更新时间                                    |
| remark       | String  | 备注信息                                    |

## 4. 相关 API

- OnsTopicCreate：创建 Topic
- OnsTopicList：查看 Topic 列表
- OnsPublishDelete：删除发布关系
- OnsSubscribeDelete：删除订阅关系

## 5. 使用示例

本示例仅提供一个参考，从华东1接入点接入，查询当前用户下所有的 Topic 列表。

```
public static void main(String[] args) {
    String regionId = "cn-hangzhou";
```

```
String accessKey = "XXXXXXXXXXXXXXXXXX";
String secretKey = "XXXXXXXXXXXXXXXXXX";
String endPointName = "cn-hangzhou";
String productName = "Ons";
String domain = "ons.cn-hangzhou.aliyuncs.com";

/**
 *根据自己需要访问的区域选择Region，并设置对应的接入点
 */
try {
    DefaultProfile.addEndpoint(endPointName, regionId, productName, domain);
} catch (ClientException e) {
    e.printStackTrace();
}
IClientProfile profile = DefaultProfile.getProfile(regionId, accessKey, secretKey);
IAcsClient IAcsClient = new DefaultAcsClient(profile);

OnsTopicSearchRequest request = new OnsTopicSearchRequest();
request.setAcceptFormat(FormatType.JSON);
/**
 *ONSRegionId是指你需要API访问MQ哪个区域的资源。
 *该值必须要根据OnsRegionList方法获取的列表来选择和配置，因为OnsRegionId是变动的，不能够写固定值
 */
request.setOnsRegionId("daily");
request.setPreventCache(System.currentTimeMillis());
request.setSearch("JODIE_TEST_28dd");
try {
    OnsTopicSearchResponse response = IAcsClient.getAcsResponse(request);
    List<OnsTopicSearchResponse.PublishInfoDo> publishInfoDoList = response.getData();
    for(OnsTopicSearchResponse.PublishInfoDo publishInfoDo:publishInfoDoList){
        System.out.println(publishInfoDo.getTopic()+" "+publishInfoDo.getOwner());
    }
    System.out.println(response.getRequestId());
} catch (ClientException e) {
    e.printStackTrace();
}
catch (Exception e) {
    e.printStackTrace();
}
}
```

本文介绍 OnsTopicStatus 接口，调用该接口可以查询指定 Topic 当前的消息状态。

## 使用场景

查询当前 Topic 下的消息总量以及最后更新时间，一般用于判断 Topic 资源的使用率。TopicStatus 接口返回当前服务器上该 Topic 下所有的消息数，以及最后消息写入时间。

## 请求参数列表

| 名称          | 类型     | 是否必须 | 描述              |
|-------------|--------|------|-----------------|
| OnsRegionId | String | 是    | 设置所要查询的区域，具体值通过 |

|              |        |   |                        |
|--------------|--------|---|------------------------|
|              |        |   | OnsRegionList接口获取      |
| OnsPlatform  | String | 否 | 该请求来源，默认是从 POP 平台      |
| PreventCache | Long   | 是 | 用于 CSRF 校验，设置为系统当前时间即可 |
| Topic        | String | 是 | 查询的 Topic 名称           |

返回参数列表

| 名称        | 类型                          | 描述                   |
|-----------|-----------------------------|----------------------|
| RequestId | String                      | 为公共参数，每个请求独一无二       |
| HelpUrl   | String                      | 帮助链接                 |
| Data      | OnsTopicStatusResponse.Data | 查询到的 Topic 的信息存储数据结构 |

OnsTopicStatusResponse.Data 数据结构

| 成员            | 类型   | 描述                     |
|---------------|------|------------------------|
| totalCount    | Long | 当前 Topic 的所有分区存在的消息数总和 |
| lastTimeStamp | Long | 当前 Topic 的最后更新时间       |

相关 API

- OnsTopicCreate：创建 Topic
- OnsConsumerStatus：查询订阅组的消费者状态

使用示例

本示例仅提供一个参考，从华东1接入点接入，查询当前用户下名为“ JODIE\_TEST\_27dd” 的 Topic 的消息数量。

```
public static void main(String[] args) {
    String regionId = "cn-hangzhou";
    String accessKey = "XXXXXXXXXXXXXXXXXX";
    String secretKey = "XXXXXXXXXXXXXXXXXX";
    String endPointName = "cn-hangzhou";
    String productName = "Ons";
    String domain = "ons.cn-hangzhou.aliyuncs.com";

    /**
     *根据自己需要访问的区域选择Region，并设置对应的接入点
     */
    try {
```

```
DefaultProfile.addEndpoint(endPointName , regionId , productName , domain);
} catch (ClientException e) {
    e.printStackTrace();
}
IClientProfile profile= DefaultProfile.getProfile(regionId , accessKey , secretKey);
IAcsClient iAcsClient= new DefaultAcsClient(profile);
OnsTopicStatusRequest request = new OnsTopicStatusRequest();
request.setAcceptFormat(FormatType.JSON);
/**
 *ONSRegionId是指你需要API访问MQ哪个区域的资源.
 *该值必须要根据OnsRegionList方法获取的列表来选择和配置，因为OnsRegionId是变动的，不能够写固定值
 */
request.setOnsRegionId("daily");
request.setPreventCache(System.currentTimeMillis());
request.setTopic("JODIE_TEST_27dd");
try {
    OnsTopicStatusResponse response = iAcsClient.getAcsResponse(request);
    OnsTopicStatusResponse.Data data=response.getData();
    Long totalCount =data.getTotalCount();
    Long lastTimeStamp =data.getLastTimeStamp();
    System.out.println(response.getRequestId());
} catch (ClientException e) {
    e.printStackTrace();
}
catch (Exception e) {
    e.printStackTrace();
}
}
```

本接口限企业铂金版客户专用，请前往铂金版购买页面查看详情。

## 使用场景

根据您的指定的 MQ 区域以及 Topic 名称，可以配置该 Topic 的读写开关。一般用于禁读或者禁写的场景。

## 主子帐号约束

本接口默认仅限主账号使用，RAM 子账号无法使用，除非该 RAM 子账号被授予了相关权限。

## 请求参数列表

| 名称           | 类型     | 是否必须 | 描述                                   |
|--------------|--------|------|--------------------------------------|
| OnsRegionId  | String | 是    | 当前查询 MQ 所在区域，可以通过 OnsRegionList 方法获取 |
| OnsPlatform  | String | 否    | 该请求来源，默认是从 POP 平台                    |
| PreventCache | Long   | 是    | 用于 CSRF 校验，设置为系统当前时间即可               |

|       |         |   |                                          |
|-------|---------|---|------------------------------------------|
| Topic | String  | 是 | 需要配置的 Topic 名称，在一个用户下不可重复，即使区域不同         |
| Perm  | Integer | 是 | 设置该 Topic 的读写模式，6 代表同时支持读写，4 代表禁写，2 代表禁读 |

返回参数列表

| 名称        | 类型     | 描述             |
|-----------|--------|----------------|
| RequestId | String | 为公共参数，每个请求独一无二 |
| HelpUrl   | String | 帮助链接           |

相关 API

- OnsTopicCreate : 创建 Topic
- OnsTopicList : 查看 Topic 列表

使用示例

本示例仅提供一个参考，从华东 1 接入点接入，设置名为“ MingduanTest” 的 Topic 为禁读模式。

```
public static void main(String[] args) {
    String regionId = "cn-hangzhou";
    String accessKey = "XXXXXXXXXXXXXXXXXX";
    String secretKey = "XXXXXXXXXXXXXXXXXX";
    String endPointName = "cn-hangzhou";
    String productName = "Ons";
    String domain = "ons.cn-hangzhou.aliyuncs.com";

    /**
    *根据自己需要访问的区域选择Region，并设置对应的接入点
    */
    try {
        DefaultProfile.addEndpoint(endPointName, regionId, productName, domain);
    } catch (ClientException e) {
        e.printStackTrace();
    }
    IClientProfile profile= DefaultProfile.getProfile(regionId, accessKey, secretKey);
    IAcsClient IAcsClient= new DefaultAcsClient(profile);

    OnsTopicUpdateRequest request = new OnsTopicUpdateRequest();
    request.setAcceptFormat(FormatType.JSON);
    request.setTopic("MingduanTest");
    /**
    *ONSRegionId是指需要API访问MQ哪个区域的资源.
    *该值必须要根据OnsRegionList方法获取的列表来选择和配置，因为OnsRegionId是变动的，不能够写固定值
    */
}
```

```
request.setOnsRegionId("daily");
request.setPreventCache(System.currentTimeMillis());
request.setPerm(4);//2代表禁读，4代表禁写，6代表同时读写
try {
    OnsTopicUpdateResponse response = IAcsClient.getAcsResponse(request);
    System.out.println(response.getRequestId());
}
catch (ServerException e) {
    e.printStackTrace();
}
catch (ClientException e) {
    e.printStackTrace();
}
catch (Exception e) {
    e.printStackTrace();
}
}
```

## 发布管理接口

本接口限企业铂金版客户专用，请前往铂金版购买页面查看详情。

### 描述

OnsPublishCreate 接口用于向 MQ 申请注册发布关系，并得到生成的 PID，用于生产环境发消息。

### 使用场景

新申请 MQ 资源，在创建 Topic 之后，需要注册对应的发布关系 (PID) 来发消息，调用 OnsPublishCreate 接口即可实现。

**注意：**每个 Topic 只需要一个 PID 即可，不需要创建多个 PID，因为PID可以在多个应用中复用。

### 请求参数

| 名称           | 类型     | 是否必须 | 描述                                             |
|--------------|--------|------|------------------------------------------------|
| OnsRegionId  | String | 是    | 当前查询 MQ 所在区域，可以通过 OnsRegionList 方法获取，请参考公共术语说明 |
| OnsPlatform  | String | 否    | 该请求来源，默认是从 POP 平台                              |
| PreventCache | Long   | 是    | 用于 CSRF 校验，设置为系统当前时间即可                         |

|            |        |   |               |
|------------|--------|---|---------------|
| ProducerId | String | 是 | 需要创建的发布关系的PID |
| Topic      | String | 是 | 发布的 Topic 名称  |

返回参数

| 名称        | 类型     | 描述             |
|-----------|--------|----------------|
| RequestId | String | 为公共参数，每个请求独一无二 |
| HelpUrl   | String | 帮助链接           |

相关 API

- OnsPublishDelete：删除发布关系
- OnsPublishList：获取发布关系列表
- OnsPublishGet：查询发布关系

使用示例

本示例从华东1接入点接入，创建一个名为 PID\_Mingduan 的发布关系到公测 Region，该发布关系表明用于发送 Topic 为 MingduanTestAPI 的消息。

```
public static void main(String []args) {
    String regionId = "cn-hangzhou";
    String accessKey = "XXXXXXXXXXXXXXXXXX";
    String secretKey = "XXXXXXXXXXXXXXXXXX";
    String endPointName = "cn-hangzhou";
    String productName ="Ons";
    String domain ="ons.cn-hangzhou.aliyuncs.com";

    /**
    *根据自己需要访问的区域选择Region，并设置对应的接入点
    */
    try {
        DefaultProfile.addEndpoint(endPointName , regionId , productName , domain);
    } catch (ClientException e) {
        e.printStackTrace();
    }
    IClientProfile profile= DefaultProfile.getProfile(regionId , accessKey , secretKey);
    IAcsClient iAcsClient= new DefaultAcsClient(profile);
    OnsPublishCreateRequest request = new OnsPublishCreateRequest();
    /**
    *ONSRegionId是指你需要API访问MQ哪个区域的资源
    *该值必须要根据OnsRegionList方法获取的列表来选择和配置，因为OnsRegionId是变动的，不能够写固定值
    */
    request.setOnsRegionId("publictest");
    request.setPreventCache(System.currentTimeMillis());
    request.setAcceptFormat(FormatType.JSON);
    request.setTopic("MingduanTestAPI");
    request.setProducerId("PID_Mingduan");
}
```

```
try {
    OnsPublishCreateResponse response=iAcsClient.getAcsResponse(request);
    System.out.println(response.getRequestId());
} catch (ServerException e) {
    e.printStackTrace();
} catch (ClientException e) {
    e.printStackTrace();
}
}
```

返回结果

CC815D34-0950-43BF-A4E8-4187C3C7BF19

本接口限企业铂金版客户专用，请前往铂金版购买页面查看详情。

描述

OnsPublishDelete接口用于删除已经发布过的 PID。

使用场景

应用下线或者资源释放时，需要删除已经存在的发布关系，调用 OnsPublishDelete 接口删除，删除前提需要确保发布关系已经存在。

请求参数

| 名称           | 类型     | 是否必须 | 描述                                   |
|--------------|--------|------|--------------------------------------|
| OnsRegionId  | String | 是    | 当前查询 MQ 所在区域，可以通过 OnsRegionList 方法获取 |
| OnsPlatform  | String | 否    | 该请求来源，默认是从 POP 平台                    |
| PreventCache | Long   | 是    | 用于 CSRF 校验，设置为系统当前时间即可               |
| ProducerId   | String | 否    | 需要删除的发布关系的 PID                       |
| Topic        | String | 是    | 需要删除的发布关系对应的 Topic                   |

返回参数

| 名称        | 类型     | 描述             |
|-----------|--------|----------------|
| RequestId | String | 为公共参数，每个请求独一无二 |
| HelpUrl   | String | 帮助链接           |

## 相关 API

- OnsPublishCreate：创建发布关系
- OnsPublishList：获取发布关系列表
- OnsPublishGet：查询发布关系

## 使用示例

以下示例用于删除日常 daily 区域的名为 PID\_MingduanTest 的 PID 关系。

```
public static void main(String []args) {
    String regionId = "cn-hangzhou";
    String accessKey = "XXXXXXXXXXXXXXXXXX";
    String secretKey = "XXXXXXXXXXXXXXXXXX";
    String endPointName = "cn-hangzhou";
    String productName = "Ons";
    String domain = "ons.cn-hangzhou.aliyuncs.com";

    /**
     *根据自己需要访问的区域选择Region，并设置对应的接入点
     */
    try {
        DefaultProfile.addEndpoint(endPointName, regionId, productName, domain);
    } catch (ClientException e) {
        e.printStackTrace();
    }
    IClientProfile profile= DefaultProfile.getProfile(regionId, accessKey, secretKey);
    IAcsClient iAcsClient= new DefaultAcsClient(profile);
    OnsPublishDeleteRequest request = new OnsPublishDeleteRequest();
    /**
     *ONSRegionId是指你需要API访问ONS哪个区域的资源
     *该值必须要根据OnsRegionList方法获取的列表来选择和配置，因为OnsRegionId是变动的，不能够写固定值
     */
    request.setOnsRegionId("daily");
    request.setPreventCache(System.currentTimeMillis());
    request.setAcceptFormat(FormatType.JSON);
    request.setTopic("Mingduan_67dd");
    request.setProducerId("PID_MingduanTest");
    try {
        OnsPublishDeleteResponse response=iAcsClient.getAcsResponse(request);
        System.out.println(response.getRequestId());
    } catch (ServerException e) {
        e.printStackTrace();
    } catch (ClientException e) {
        e.printStackTrace();
    }
}
```

## 发布信息查询

OnsPublishGet 接口用于查询指定 PID 的详细信息。

## 使用场景

调用 OnsPublishGet 接口一般用于发布信息的展示和查询。

## 请求参数列表

| 名称           | 类型     | 是否必须 | 描述                                   |
|--------------|--------|------|--------------------------------------|
| OnsRegionId  | String | 是    | 当前查询 MQ 所在区域，可以通过 OnsRegionList 方法获取 |
| OnsPlatform  | String | 否    | 该请求来源，默认是从 POP 平台                    |
| PreventCache | Long   | 是    | 用于 CSRF 校验，设置为系统当前时间即可               |
| ProducerId   | String | 是    | 需要查询的发布关系的 PID                       |
| Topic        | String | 是    | 需要查询的 Topic                          |

## 返回参数列表

| 名称        | 类型                  | 描述             |
|-----------|---------------------|----------------|
| RequestId | String              | 为公共参数，每个请求独一无二 |
| HelpUrl   | String              | 帮助链接           |
| Data      | List(PublishInfoDo) | 查询结果集合         |

## PublishInfoDo 数据结构

| 成员          | 类型      | 描述                                          |
|-------------|---------|---------------------------------------------|
| id          | Long    | 该发布信息在数据库中的 ID                              |
| channelId   | Integer | 该 Topic 所在区域 ID，0-ALIYUN，1-CLOUD，2，3，4      |
| channelName | String  | 该 Topic 所在区域名称，ALIYUN 代表阿里云，CLOUD 代表聚石塔，... |
| onsRegionId | Long    | 该 Topic 所在区域 ID，就是 ONSRegionList 方法获取的内容    |
| regionName  | String  | 该 Topic 所在区域名称                              |
| topic       | String  | Topic 名称                                    |
| owner       | String  | Topic 所有者编号                                 |
| relation    | Integer | 所有关系编号1为持有者，2为                              |

|              |         |                            |
|--------------|---------|----------------------------|
|              |         | 可以发布，4为可以订阅，6为可以发布和订阅      |
| relationName | String  | 所有关系名称，例如持有者、可订阅、可发布、可发布订阅 |
| status       | Integer | 当前状态编号0—服务中                |
| statusName   | String  | 当前状态别名，服务中，                |
| appkey       | Integer | null                       |
| createTime   | Long    | 创建时间                       |
| updateTime   | Long    | 更新时间                       |
| remark       | String  | 备注，可以不填                    |

相关 API

- OnsPublishDelete：删除发布关系
- OnsPublishList：获取发布关系列表
- OnsPublishCreate：新建发布关系

使用示例

以下示例用于查询 daily 区域名为 PID\_Mingduan 的 PID 信息，并打印主要属性。

```
public static void main(String []args) {
    String regionId = "cn-hangzhou";
    String accessKey = "XXXXXXXXXXXXXXXXXX";
    String secretKey = "XXXXXXXXXXXXXXXXXX";
    String endPointName = "cn-hangzhou";
    String productName = "Ons";
    String domain = "ons.cn-hangzhou.aliyuncs.com";

    /**
    *根据自己需要访问的区域选择Region，并设置对应的接入点
    */
    try {
        DefaultProfile.addEndpoint(endPointName , regionId , productName , domain);
    } catch (ClientException e) {
        e.printStackTrace();
    }
    IClientProfile profile= DefaultProfile.getProfile(regionId , accessKey , secretKey);
    IAcsClient iAcsClient= new DefaultAcsClient(profile);
    OnsPublishGetRequest request = new OnsPublishGetRequest();
    /**
    *ONSRegionId是指你需要API访问MQ哪个区域的资源.
    *该值必须要根据OnsRegionList方法获取的列表来选择和配置，因为OnsRegionId是变动的，不能够写固定值
    */
    request.setOnsRegionId("daily");
    request.setPreventCache(System.currentTimeMillis());
    request.setAcceptFormat(FormatType.JSON);
}
```

```
request.setTopic("MingduanTest");
request.setProducerId("PID_Mingduan");
try {
    OnsPublishGetResponse response=iAcsClient.getAcsResponse(request);
    List<OnsPublishGetResponse.PublishInfoDo> publishInfoDoList =response.getData();
    for (OnsPublishGetResponse.PublishInfoDo publishInfoDo:publishInfoDoList){
        System.out.println(publishInfoDo.getId()+" "+
            publishInfoDo.getChannelId()+" "+
            publishInfoDo.getChannelName()+" "+
            publishInfoDo.getOnsRegionId()+" "+
            publishInfoDo.getRegionName()+" "+
            publishInfoDo.getOwner()+" "+
            publishInfoDo.getProducerId()+" "+
            publishInfoDo.getTopic()+" "+
            publishInfoDo.getStatus()+" "+
            publishInfoDo.getStatusName()+" "+
            publishInfoDo.getCreateTime()+" "+
            publishInfoDo.getUpdateTime());
    }
} catch (ServerException e) {
    e.printStackTrace();
} catch (ClientException e) {
    e.printStackTrace();
}
}
```

OnsPublishList 接口用于获取指定用户所拥有的所有发布关系。

使用场景

OnsPublishList 接口一般用于展示用户所拥有的发布关系列表。

请求参数列表

| 名称           | 类型     | 是否必须 | 描述                               |
|--------------|--------|------|----------------------------------|
| OnsRegionId  | String | 是    | 设置所要查询的区域，具体值通过OnsRegionList接口获取 |
| OnsPlatform  | String | 否    | 该请求来源，默认是从POP 平台                 |
| PreventCache | Long   | 是    | 用于 CSRF 校验，设置为系统当前时间即可           |

返回参数

| 名称        | 类型     | 描述             |
|-----------|--------|----------------|
| RequestId | String | 为公共参数，每个请求独一无二 |
| HelpUrl   | String | 帮助链接           |

|      |                     |        |
|------|---------------------|--------|
| Data | List(PublishInfoDo) | 查询结果集合 |
|------|---------------------|--------|

PublishInfoDo 数据结构

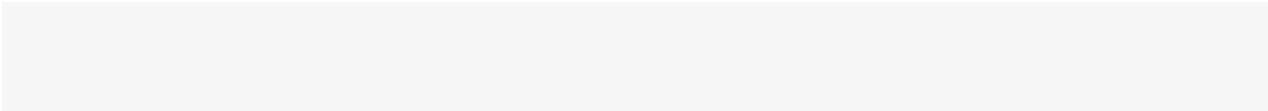
| 成员           | 类型      | 描述                                          |
|--------------|---------|---------------------------------------------|
| id           | Long    | 该发布信息在数据库中的 ID                              |
| channelId    | Integer | 该 Topic 所在区域 ID，0-ALIYUN，1-CLOUD，2，3，4      |
| channelName  | String  | 该 Topic 所在区域名称，ALIYUN 代表阿里云，CLOUD 代表聚石塔，... |
| onsRegionId  | Long    | 该 Topic 所在区域 ID，就是ONSRegionList 方法获取的内容     |
| regionName   | String  | 该 Topic 所在区域名称                              |
| topic        | String  | Topic 名称                                    |
| owner        | String  | Topic 所有者编号                                 |
| relation     | Integer | 所有关系编号1为持有者，2为可以发布，4为可以订阅，6为可以发布和订阅         |
| relationName | String  | 所有关系名称，例如持有者、可订阅、可发布、可发布订阅                  |
| status       | Integer | 当前状态编号0-服务中                                 |
| statusName   | String  | 当前状态别名，服务中                                  |
| appkey       | Integer | null                                        |
| createTime   | Long    | 创建时间                                        |
| updateTime   | Long    | 更新时间                                        |
| remark       | String  | 备注，可以不填                                     |

相关 API

- OnsPublishDelete：删除发布关系
- OnsPublishGet：查询发布关系
- OnsPublishCreate：新建发布关系

使用示例

本示例用于查询用户在 daily 区域下的所有 PID 的列表，并打印。



```
public static void main(String []args) {
    String regionId = "cn-hangzhou";
    String accessKey = "XXXXXXXXXXXXXXXXXX";
    String secretKey = "XXXXXXXXXXXXXXXXXX";
    String endPointName = "cn-hangzhou";
    String productName = "Ons";
    String domain = "ons.cn-hangzhou.aliyuncs.com";

    /**
     *根据自己需要访问的区域选择Region，并设置对应的接入点
     */
    try {
        DefaultProfile.addEndpoint(endPointName, regionId, productName, domain);
    } catch (ClientException e) {
        e.printStackTrace();
    }
    IClientProfile profile= DefaultProfile.getProfile(regionId, accessKey, secretKey);
    IAcsClient iAcsClient= new DefaultAcsClient(profile);
    OnsPublishListRequest request = new OnsPublishListRequest();
    /**
     *ONSRegionId是指你需要API访问MQ哪个区域的资源。
     *该值必须要根据OnsRegionList方法获取的列表来选择和配置，因为OnsRegionId是变动的，不能够写固定值
     */
    request.setOnsRegionId("daily");
    request.setPreventCache(System.currentTimeMillis());
    request.setAcceptFormat(FormatType.JSON);
    try {
        OnsPublishListResponse response=iAcsClient.getAcsResponse(request);
        List<OnsPublishListResponse.PublishInfoDo> publishInfoDoList =response.getData();
        for (OnsPublishListResponse.PublishInfoDo publishInfoDo:publishInfoDoList){
            System.out.println(publishInfoDo.getId()+" "+
                publishInfoDo.getChannelId()+" "+
                publishInfoDo.getChannelName()+" "+
                publishInfoDo.getOnsRegionId()+" "+
                publishInfoDo.getRegionName()+" "+
                publishInfoDo.getOwner()+" "+
                publishInfoDo.getProducerId()+" "+
                publishInfoDo.getTopic()+" "+
                publishInfoDo.getStatus()+" "+
                publishInfoDo.getStatusName()+" "+
                publishInfoDo.getCreateTime()+" "+
                publishInfoDo.getUpdateTime());
        }
    } catch (ServerException e) {
        e.printStackTrace();
    } catch (ClientException e) {
        e.printStackTrace();
    }
}
```

本文介绍 OnsPublishSearch 接口，调用该接口根据搜索条件查询相关的发布关系。

## 1. 使用场景

OnsPublishSearch 接口一般用在用户账号下 PID 太多无法同时展现时，根据关键字搜索符合条件的发布关系，展现的数据格式和 OnsPublishGet 接口一致。

## 2. 请求参数列表

| 名称           | 类型     | 是否必须 | 描述                                   |
|--------------|--------|------|--------------------------------------|
| OnsRegionId  | String | 是    | 当前查询 MQ 所在区域，可以通过 OnsRegionList 方法获取 |
| OnsPlatform  | String | 否    | 该请求来源，默认是从 POP 平台                    |
| PreventCache | Long   | 是    | 用于 CSRF 校验，设置为系统当前时间即可               |
| Search       | String | 是    | 搜索的关键词，可以是 Topic 或者 PID              |

## 3. 返回参数列表

| 名称        | 类型                  | 描述             |
|-----------|---------------------|----------------|
| RequestId | String              | 为公共参数，每个请求独一无二 |
| HelpUrl   | String              | 帮助链接           |
| Data      | List(PublishInfoDo) | 查询结果集合         |

### PublishInfoDo 数据结构

| 成员          | 类型      | 描述                                          |
|-------------|---------|---------------------------------------------|
| id          | Long    | 该发布信息在数据库中的 ID                              |
| channelId   | Integer | 该 Topic 所在区域 ID，0-ALIYUN，1-CLOUD，2，3，4      |
| channelName | String  | 该 Topic 所在区域名称，ALIYUN 代表阿里云，CLOUD 代表聚石塔，... |
| onsRegionId | Long    | 该 Topic 所在区域 ID，就是 ONSRegionList 方法获取的内容    |
| regionName  | String  | 该 Topic 所在区域名称                              |
| topic       | String  | Topic 名称                                    |
| owner       | String  | Topic 所有者编号                                 |
| relation    | Integer | 所有关系编号1为持有者，2为可以发布，4为可以订阅，6为可以发布和订阅         |

|              |         |                            |
|--------------|---------|----------------------------|
| relationName | String  | 所有关系名称，例如持有者、可订阅、可发布、可发布订阅 |
| status       | Integer | 当前状态编号0-服务中                |
| statusName   | String  | 当前状态别名，服务中                 |
| appkey       | Integer | null                       |
| createTime   | Long    | 创建时间                       |
| updateTime   | Long    | 更新时间                       |
| remark       | String  | 备注，可以不填                    |

## 4. 相关 API

- OnsPublishDelete：删除发布关系
- OnsPublishGet：查询发布关系
- OnsPublishCreate：新建发布关系

## 5. 使用示例

### 调用示例

本示例以 PID\_Mingduan 为搜索条件搜索 daily 区域下相关的 PID 信息，并将结果打印。

```
public static void main(String []args) {
    String regionId = "cn-hangzhou";
    String accessKey = "XXXXXXXXXXXXXXXXXX";
    String secretKey = "XXXXXXXXXXXXXXXXXX";
    String endPointName = "cn-hangzhou";
    String productName = "Ons";
    String domain = "ons.cn-hangzhou.aliyuncs.com";

    /**
     *根据自己需要访问的区域选择Region，并设置对应的接入点
     */
    try {
        DefaultProfile.addEndpoint(endPointName, regionId, productName, domain);
    } catch (ClientException e) {
        e.printStackTrace();
    }
    IClientProfile profile= DefaultProfile.getProfile(regionId, accessKey, secretKey);
    IAcsClient iAcsClient= new DefaultAcsClient(profile);
    OnsPublishSearchRequest request = new OnsPublishSearchRequest();
    /**
     *ONSRegionId是指你需要API访问MQ哪个区域的资源.
     *该值必须要根据OnsRegionList方法获取的列表来选择和配置，因为OnsRegionId是变动的，不能够写固定值
     */
    request.setOnsRegionId("daily");
    request.setPreventCache(System.currentTimeMillis());
    request.setAcceptFormat(FormatType.JSON);
}
```

```
request.setSearch("PID_Mingduan");
try {
    OnsPublishSearchResponse response=iAcsClient.getAcsResponse(request);
    List<OnsPublishSearchResponse.PublishInfoDo> publishInfoDoList =response.getData();
    for (OnsPublishSearchResponse.PublishInfoDo publishInfoDo:publishInfoDoList){
        System.out.println(publishInfoDo.getId()+" "+
            publishInfoDo.getChannelId()+" "+
            publishInfoDo.getChannelName()+" "+
            publishInfoDo.getOnsRegionId()+" "+
            publishInfoDo.getRegionName()+" "+
            publishInfoDo.getOwner()+" "+
            publishInfoDo.getProducerId()+" "+
            publishInfoDo.getTopic()+" "+
            publishInfoDo.getStatus()+" "+
            publishInfoDo.getStatusName()+" "+
            publishInfoDo.getCreateTime()+" "+
            publishInfoDo.getUpdateTime());
    }
} catch (ServerException e) {
    e.printStackTrace();
} catch (ClientException e) {
    e.printStackTrace();
}
}
```

## 订阅管理接口

本接口限企业铂金版客户专用，请前往铂金版购买页面查看详情。

### 描述

OnsSubscribeCreate 接口用于在服务器上创建对特定 Topic 的订阅关系。

### 使用场景

新应用上线时，在注册 Topic 资源后，如果需要订阅消息，需要首先在 MQ 控制台或者调用该 API 申请注册对目标 Topic 的注册关系，然后才能使用注册的 CID 订阅消息。

### 请求参数

| 名称          | 类型     | 是否必须 | 描述                                   |
|-------------|--------|------|--------------------------------------|
| OnsRegionId | String | 是    | 当前查询 MQ 所在区域，可以通过 OnsRegionList 方法获取 |
| OnsPlatform | String | 否    | 该请求来源，默认是从                           |

|              |        |   |                        |
|--------------|--------|---|------------------------|
|              |        |   | POP 平台                 |
| PreventCache | Long   | 是 | 用于 CSRF 校验，设置为系统当前时间即可 |
| ConsumerId   | String | 是 | 创建的消息消费集群的 CID         |
| Topic        | String | 是 | 订阅的 Topic              |

返回参数

| 名称        | 类型     | 描述             |
|-----------|--------|----------------|
| RequestId | String | 为公共参数，每个请求独一无二 |
| HelpUrl   | String | 帮助链接           |

相关 API

- OnsSubscribeDelete：删除订阅关系
- OnsSubscribeList：查询订阅关系列表
- OnsSubscribeGet：精确查询订阅关系信息

使用示例

本示例在 daily 区域创建名为 CID\_MingduanTest 的订阅关系用于消费 Mingduan\_67dd 的消息。

```
public static void main(String []args) {
String regionId = "cn-hangzhou";
String accessKey = "XXXXXXXXXXXXXXXXXX";
String secretKey = "XXXXXXXXXXXXXXXXXX";
String endPointName = "cn-hangzhou";
String productName ="Ons";
String domain ="ons.cn-hangzhou.aliyuncs.com";

/**
*根据自己需要访问的区域选择Region，并设置对应的接入点
*/
try {
DefaultProfile.addEndpoint(endPointName, regionId, productName, domain);
} catch (ClientException e) {
e.printStackTrace();
}
IClientProfile profile= DefaultProfile.getProfile(regionId, accessKey, secretKey);
IAcsClient iAcsClient= new DefaultAcsClient(profile);
OnsSubscriptionCreateRequest request = new OnsSubscriptionCreateRequest();
/**
*ONSRegionId是指你需要API访问ONS哪个区域的资源
*该值必须要根据OnsRegionList方法获取的列表来选择和配置，因为OnsRegionId是变动的，不能够写固定值
*/
request.setOnsRegionId("daily");
request.setPreventCache(System.currentTimeMillis());
}
```

```
request.setAcceptFormat(FormatType.JSON);
request.setTopic("Mingduan_67dd");
request.setConsumerId("CID_MingduanTest");
try {
    OnsSubscriptionCreateResponse response=iAcsClient.getAcsResponse(request);
    System.out.println(response.getRequestId());
} catch (ServerException e) {
    e.printStackTrace();
} catch (ClientException e) {
    e.printStackTrace();
}
}
```

本接口限企业铂金版客户专用，请前往铂金版购买页面查看详情。

描述

OnsSubscribeDelete 接口用于删除之前使用 OnsSubscribeCretae 接口创建的订阅关系。

使用场景

应用下线时销毁 MQ 的资源可以调用 OnsSubscribeDelete 接口删除订阅关系信息。

请求参数

| 名称           | 类型     | 是否必须 | 描述                                   |
|--------------|--------|------|--------------------------------------|
| OnsRegionId  | String | 是    | 当前查询 MQ 所在区域，可以通过 OnsRegionList 方法获取 |
| OnsPlatform  | String | 否    | 该请求来源，默认是从 POP 平台                    |
| PreventCache | Long   | 是    | 用于 CSRF 校验，设置为系统当前时间即可               |
| ConsumerId   | String | 是    | 需要删除的消息消费集群的 CID                     |
| Topic        | String | 是    | 订阅的 Topic                            |

返回参数

| 名称        | 类型     | 描述             |
|-----------|--------|----------------|
| RequestId | String | 为公共参数，每个请求独一无二 |
| HelpUrl   | String | 帮助链接           |

相关接口

- OnsSubscribberCreate：创建订阅关系
- OnsSubscribeList：查询订阅关系列表

## 使用示例

```
public static void main(String []args) {
    String regionId = "cn-hangzhou";
    String accessKey = "XXXXXXXXXXXXXXXXXX";
    String secretKey = "XXXXXXXXXXXXXXXXXX";
    String endPointName = "cn-hangzhou";
    String productName = "Ons";
    String domain = "ons.cn-hangzhou.aliyuncs.com";

    /**
     *根据自己需要访问的区域选择Region，并设置对应的接入点
     */
    try {
        DefaultProfile.addEndpoint(endPointName, regionId, productName, domain);
    } catch (ClientException e) {
        e.printStackTrace();
    }
    IClientProfile profile= DefaultProfile.getProfile(regionId, accessKey, secretKey);
    IAcsClient iAcsClient= new DefaultAcsClient(profile);
    OnsSubscriptionDeleteRequest request = new OnsSubscriptionDeleteRequest();
    /**
     *ONSRegionId是指你需要API访问ONS哪个区域的资源
     *该值必须要根据OnsRegionList方法获取的列表来选择和配置，因为OnsRegionId是变动的，不能够写固定值
     */
    request.setOnsRegionId("daily");
    request.setPreventCache(System.currentTimeMillis());
    request.setAcceptFormat(FormatType.JSON);
    request.setTopic("Mingduan_67dd");
    request.setConsumerId("CID_MingduanTest");
    try {
        OnsSubscriptionDeleteResponse response=iAcsClient.getAcsResponse(request);
        System.out.println(response.getRequestId());
    } catch (ServerException e) {
        e.printStackTrace();
    } catch (ClientException e) {
        e.printStackTrace();
    }
}
```

OnsSubscribeGet 接口用于查询目标 CID 的详细信息。

## 使用场景

MQ 使用过程中如果需要对订阅关系进行管理，获取账号下 CID 的详细信息，可以调用该接口获取信息。

## 请求参数列表

| 名称 | 类型 | 是否必须 | 描述 |
|----|----|------|----|
|----|----|------|----|

|              |        |   |                                    |
|--------------|--------|---|------------------------------------|
| OnsRegionId  | String | 是 | 设置所要查询的区域，具体值通过 OnsRegionList 接口获取 |
| OnsPlatform  | String | 否 | 该请求来源，默认是从 POP 平台                  |
| PreventCache | Long   | 是 | 用于 CSRF 校验，设置为系统当前时间即可             |
| ConsumerId   | String | 是 | 需要查询的订阅关系的 CID                     |
| Topic        | String | 是 | 需要查询的 Topic                        |

## 返回参数列表

| 名称        | 类型                    | 描述             |
|-----------|-----------------------|----------------|
| RequestId | String                | 为公共参数，每个请求独一无二 |
| HelpUrl   | String                | 帮助链接           |
| Data      | List(SubscribeInfoDo) | 查询到的订阅关系集合     |

## SubscribeInfoDo 数据结构

| 名称          | 类型      | 描述                                          |
|-------------|---------|---------------------------------------------|
| Id          | Long    | 订阅信息在数据库中的索引编号                              |
| channelId   | Integer | 该 Topic 所在区域 ID，0-ALIYUN，1-CLOUD，2，3，4      |
| channelName | String  | 该 Topic 所在区域名称，ALIYUN 代表阿里云，CLOUD 代表聚石塔，... |
| onsRegionId | Long    | 该 Topic 所在区域 ID，就是 ONSRegionList 方法获取的内容    |
| regionName  | String  | 该 Topic 所在区域名称                              |
| Owner       | String  | 该发布信息的拥有者                                   |
| ConsumerId  | String  | 消费集群 ID                                     |
| Topic       | String  | Topic 名称                                    |
| Status      | Long    | 当前状态                                        |
| StatusName  | String  | 状态名称，服务中，已下线等                               |
| CreateTime  | Long    | 创建时间                                        |
| UpdateTime  | Long    | 最后更新时间                                      |

## 相关 API

- OnsSubscribeCreate：创建订阅关系
- OnsSubscribeDelete：删除订阅关系
- OnsSubscribeList：获取订阅关系列表

## 使用示例

本示例在 daily 区域下查询名为 CID\_Mingduan 的订阅组信息，并打印。

```
public static void main(String []args) {
    String regionId = "cn-hangzhou";
    String accessKey = "XXXXXXXXXXXXXXXXXX";
    String secretKey = "XXXXXXXXXXXXXXXXXX";
    String endPointName = "cn-hangzhou";
    String productName = "Ons";
    String domain = "ons.cn-hangzhou.aliyuncs.com";

    /**
     *根据自己需要访问的区域选择Region，并设置对应的接入点
     */
    try {
        DefaultProfile.addEndpoint(endPointName, regionId, productName, domain);
    } catch (ClientException e) {
        e.printStackTrace();
    }
    IClientProfile profile= DefaultProfile.getProfile(regionId, accessKey, secretKey);
    IAcsClient iAcsClient= new DefaultAcsClient(profile);
    OnsSubscriptionGetRequest request = new OnsSubscriptionGetRequest();
    /**
     *ONSRegionId是指你需要API访问MQ哪个区域的资源
     *该值必须要根据OnsRegionList方法获取的列表来选择和配置，因为OnsRegionId是变动的，不能够写固定值
     */
    request.setOnsRegionId("daily");
    request.setPreventCache(System.currentTimeMillis());
    request.setAcceptFormat(FormatType.JSON);
    request.setTopic("MingduanTest");
    request.setConsumerId("CID_Mingduan");
    try {
        OnsSubscriptionGetResponse response=iAcsClient.getAcsResponse(request);
        List<OnsSubscriptionGetResponse.SubscribeInfoDo> subscribeInfoDoList=response.getData();
        for(OnsSubscriptionGetResponse.SubscribeInfoDo subscribeInfoDo:subscribeInfoDoList){
            System.out.println(subscribeInfoDo.getId()+" "+
                subscribeInfoDo.getChannelId()+" "+
                subscribeInfoDo.getChannelName()+" "+
                subscribeInfoDo.getOnsRegionId()+" "+
                subscribeInfoDo.getRegionName()+" "+
                subscribeInfoDo.getOwner()+" "+
                subscribeInfoDo.getConsumerId()+" "+
                subscribeInfoDo.getTopic()+" "+
                subscribeInfoDo.getStatus()+" "+
                subscribeInfoDo.getStatusName()+" "+
                subscribeInfoDo.getCreateTime()+" "+
```

```
subscribeInfoDo.getUpdateTime());
}
} catch (ServerException e) {
e.printStackTrace();
} catch (ClientException e) {
e.printStackTrace();
}
}
```

OnsSubscribeList 接口用于获取用户所有的 CID 信息的列表。

使用场景

OnsSubscribeList 接口一般用于用户 CID 资源的列表展示，并不用于具体的信息查询。

请求参数列表

| 名称           | 类型     | 是否必须 | 描述                                |
|--------------|--------|------|-----------------------------------|
| OnsRegionId  | String | 是    | 设置所要查询的区域，具体值通过 OnsRegionList接口获取 |
| OnsPlatform  | String | 否    | 该请求来源，默认是从 POP 平台                 |
| PreventCache | Long   | 是    | 用于 CSRF 校验，设置为系统当前时间即可            |

返回参数列表

| 名称        | 类型                    | 描述             |
|-----------|-----------------------|----------------|
| RequestId | String                | 为公共参数，每个请求独一无二 |
| HelpUrl   | String                | 帮助链接           |
| Data      | List(SubscribeInfoDo) | 查询到的订阅关系集合     |

SubscribeInfoDo 数据结构

| 名称          | 类型      | 描述                                          |
|-------------|---------|---------------------------------------------|
| Id          | Long    | 订阅信息在数据库中的索引编号                              |
| channelId   | Integer | 该 Topic 所在区域 ID，0-ALIYUN，1-CLOUD，2，3，4      |
| channelName | String  | 该 Topic 所在区域名称，ALIYUN 代表阿里云，CLOUD 代表聚石塔，... |
| onsRegionId | Long    | 该 Topic 所在区域 ID，就是                          |

|            |        | ONSRegionList 方法获取的内容 |
|------------|--------|-----------------------|
| regionName | String | 该 Topic 所在区域名称        |
| Owner      | String | 该发布信息的拥有者             |
| ConsumerId | String | 消费集群 ID               |
| Topic      | String | Topic 名称              |
| Status     | Long   | 当前状态                  |
| StatusName | String | 状态名称，服务中，已下线等         |
| CreateTime | Long   | 创建时间                  |
| UpdateTime | Long   | 最后更新时间                |

相关 API

- OnsSubscribeCreate：创建订阅关系
- OnsSubscribeDelete：删除订阅关系
- OnsSubscribeGet：订阅关系精确查询

使用示例

本示例查询 daily 区域下所有的 CID 列表，并打印。

```
public static void main(String []args) {
    String regionId = "cn-hangzhou";
    String accessKey = "XXXXXXXXXXXXXXXXXX";
    String secretKey = "XXXXXXXXXXXXXXXXXX";
    String endPointName = "cn-hangzhou";
    String productName = "Ons";
    String domain = "ons.cn-hangzhou.aliyuncs.com";

    /**
    *根据自己需要访问的区域选择Region，并设置对应的接入点
    */
    try {
        DefaultProfile.addEndpoint(endPointName, regionId, productName, domain);
    } catch (ClientException e) {
        e.printStackTrace();
    }
    IClientProfile profile= DefaultProfile.getProfile(regionId, accessKey, secretKey);
    IAcsClient iAcsClient= new DefaultAcsClient(profile);
    OnsSubscriptionListRequest request = new OnsSubscriptionListRequest();
    /**
    *ONSRegionId是指你需要API访问MQ哪个区域的资源.
    *该值必须要根据OnsRegionList方法获取的列表来选择和配置，因为OnsRegionId是变动的，不能够写固定值
    */
    request.setOnsRegionId("daily");
    request.setPreventCache(System.currentTimeMillis());
}
```

```
request.setAcceptFormat(FormatType.JSON);
try {
    OnsSubscriptionListResponse response=iAcsClient.getAcsResponse(request);
    List<OnsSubscriptionListResponse.SubscribeInfoDo> subscribeInfoDoList=response.getData();
    for(OnsSubscriptionListResponse.SubscribeInfoDo subscribeInfoDo:subscribeInfoDoList){
        System.out.println(subscribeInfoDo.getId()+" "+
            subscribeInfoDo.getChannelId()+" "+
            subscribeInfoDo.getChannelName()+" "+
            subscribeInfoDo.getOnsRegionId()+" "+
            subscribeInfoDo.getRegionName()+" "+
            subscribeInfoDo.getOwner()+" "+
            subscribeInfoDo.getConsumerId()+" "+
            subscribeInfoDo.getTopic()+" "+
            subscribeInfoDo.getStatus()+" "+
            subscribeInfoDo.getStatusName()+" "+
            subscribeInfoDo.getCreateTime()+" "+
            subscribeInfoDo.getUpdateTime());
    }
} catch (ServerException e) {
    e.printStackTrace();
} catch (ClientException e) {
    e.printStackTrace();
}
}
```

OnsSubscribeSearch 接口通过 Topic 或者 CID 信息搜索符合条件的 CID 信息。

## 使用场景

OnsSubscribeSearch 接口一般用于不确定 CID 名称的时候，可以通过 Topic 来搜索 CID 信息。

## 请求参数列表

| 名称           | 类型     | 是否必须 | 描述                                   |
|--------------|--------|------|--------------------------------------|
| OnsRegionId  | String | 是    | 当前查询 MQ 所在区域，可以通过 OnsRegionList 方法获取 |
| OnsPlatform  | String | 否    | 该请求来源，默认是从 POP 平台                    |
| PreventCache | Long   | 是    | 用于 CSRF 校验，设置为系统当前时间即可               |
| Search       | String | 是    | 搜索的关键词，可以是 Topic 或者 CID              |

## 返回参数列表

| 名称        | 类型     | 描述             |
|-----------|--------|----------------|
| RequestId | String | 为公共参数，每个请求独一无二 |

|         |                       |            |
|---------|-----------------------|------------|
| HelpUrl | String                | 帮助链接       |
| Data    | List(SubscribeInfoDo) | 查询到的订阅关系集合 |

SubscribeInfoDo 数据结构

| 名称          | 类型      | 描述                                          |
|-------------|---------|---------------------------------------------|
| Id          | Long    | 订阅信息在数据库中的索引编号                              |
| channelId   | Integer | 该 Topic 所在区域 ID，0-ALIYUN，1-CLOUD，2，3，4      |
| channelName | String  | 该 Topic 所在区域名称，ALIYUN 代表阿里云，CLOUD 代表聚石塔，... |
| onsRegionId | Long    | 该 Topic 所在区域 ID，就是ONSRegionList 方法获取的内容     |
| regionName  | String  | 该 Topic 所在区域名称                              |
| Owner       | String  | 该发布信息的拥有者                                   |
| ConsumerId  | String  | 消费集群 ID                                     |
| Topic       | String  | Topic 名称                                    |
| Status      | Long    | 当前状态                                        |
| StatusName  | String  | 状态名称，服务中，已下线等                               |
| CreateTime  | Long    | 创建时间                                        |
| UpdateTime  | Long    | 最后更新时间                                      |

相关 API

- OnsSubscribeCreate：创建订阅关系
- OnsSubscribeDelete：删除订阅关系
- OnsSubscribeList：获取订阅关系列表

使用示例

本示例通过设定搜索条件 CID\_Mingduan 来获取符合的 CID 信息，并打印。

```
public static void main(String []args) {
    String regionId = "cn-hangzhou";
    String accessKey = "XXXXXXXXXXXXXXXXXX";
    String secretKey = "XXXXXXXXXXXXXXXXXX";
    String endPointName = "cn-hangzhou";
    String productName = "Ons";
    String domain = "ons.cn-hangzhou.aliyuncs.com";
```

```
/**
 *根据自己需要访问的区域选择Region，并设置对应的接入点
 */
try {
    DefaultProfile.addEndpoint(endPointName, regionId, productName, domain);
} catch (ClientException e) {
    e.printStackTrace();
}
IClientProfile profile= DefaultProfile.getProfile(regionId, accessKey, secretKey);
IAcsClient iAcsClient= new DefaultAcsClient(profile);
OnsSubscriptionSearchRequest request = new OnsSubscriptionSearchRequest();
/**
 *ONSRegionId是指你需要API访问MQ哪个区域的资源。
 *该值必须要根据OnsRegionList方法获取的列表来选择和配置，因为OnsRegionId是变动的，不能够写固定值
 */
request.setOnsRegionId("daily");
request.setPreventCache(System.currentTimeMillis());
request.setAcceptFormat(FormatType.JSON);
request.setSearch("CID_Mingduan");
try {
    OnsSubscriptionSearchResponse response=iAcsClient.getAcsResponse(request);
    List<OnsSubscriptionSearchResponse.SubscribeInfoDo> subscribeInfoDoList=response.getData();
    for(OnsSubscriptionSearchResponse.SubscribeInfoDo subscribeInfoDo:subscribeInfoDoList){
        System.out.println(subscribeInfoDo.getId()+" "+
            subscribeInfoDo.getChannelId()+" "+
            subscribeInfoDo.getChannelName()+" "+
            subscribeInfoDo.getOnsRegionId()+" "+
            subscribeInfoDo.getRegionName()+" "+
            subscribeInfoDo.getOwner()+" "+
            subscribeInfoDo.getConsumerId()+" "+
            subscribeInfoDo.getTopic()+" "+
            subscribeInfoDo.getStatus()+" "+
            subscribeInfoDo.getStatusName()+" "+
            subscribeInfoDo.getCreateTime()+" "+
            subscribeInfoDo.getUpdateTime());
    }
} catch (ServerException e) {
    e.printStackTrace();
} catch (ClientException e) {
    e.printStackTrace();
}
}
```

本接口限企业铂金版客户专用，请前往铂金版购买页面查看详情。

## 描述

OnsSubscribeUpdate 接口用于设置指定订阅组的读取消息的权限。

## 使用场景

根据您指定的 MQ 区域以及 Consumer ID 名称，可以配置该 Consumer ID 的读消息的开关，一般用于禁止

特定 Consumer ID 读取消息的场景。

请求参数列表

| 名称           | 类型      | 是否必须 | 描述                                   |
|--------------|---------|------|--------------------------------------|
| OnsRegionId  | String  | 是    | 当前查询 MQ 所在区域，可以通过 OnsRegionList 方法获取 |
| OnsPlatform  | String  | 否    | 该请求来源，默认是从 POP 平台                    |
| PreventCache | Long    | 是    | 用于 CSRF 校验，设置为系统当前时间即可               |
| ConsumerId   | String  | 是    | 需要配置的 consumer ID 名称                 |
| ReadEnable   | Boolean | 是    | 设置读消息开关                              |

返回参数列表

| 名称        | 类型     | 描述             |
|-----------|--------|----------------|
| RequestId | String | 为公共参数，每个请求独一无二 |
| HelpUrl   | String | 帮助链接           |

相关 API

- OnsSubscribeCreate：创建订阅关系
- OnsSubscribeDelete：删除订阅关系
- OnsSubscribeGet：订阅关系精确查询

使用示例

本示例仅提供一个参考，从华东 1 接入点接入，关闭名为“ CID\_001” 的 Consumer ID 的读权限。

```
public static void main(String[] args) {
    String regionId = "cn-hangzhou";
    String accessKey = "XXXXXXXXXXXXXXXXXX";
    String secretKey = "XXXXXXXXXXXXXXXXXX";
    String endPointName = "cn-hangzhou";
    String productName = "Ons";
    String domain = "ons.cn-hangzhou.aliyuncs.com";

    /**
     *根据自己需要访问的区域选择Region，并设置对应的接入点
     */
    try {
```

```
DefaultProfile.addEndpoint(endPointName , regionId , productName , domain);
} catch (ClientException e) {
e.printStackTrace();
}
IClientProfile profile= DefaultProfile.getProfile(regionId , accessKey , secretKey);
IAcsClient IAcsClient= new DefaultAcsClient(profile);

OnsSubscriptionUpdateRequest request = new OnsSubscriptionUpdateRequest();
request.setAcceptFormat(FormatType.JSON);
request.setConsumerId("CID_001");
/**
 *ONSRegionId是指需要API访问MQ哪个区域的资源
 *该值必须要根据OnsRegionList方法获取的列表来选择和配置，因为OnsRegionId是变动的，不能够写固定值
 */
request.setOnsRegionId("daily");
request.setPreventCache(System.currentTimeMillis());
request.setReadEnable(false);//2代表禁读，4代表禁写，6代表同时读写
try {
OnsSubscriptionUpdateResponse response = IAcsClient.getAcsResponse(request);
System.out.println(response.getRequestId());
}
catch (ServerException e) {
e.printStackTrace();
}
catch (ClientException e) {
e.printStackTrace();
}
catch (Exception e) {
e.printStackTrace();
}
}
```

## 消费管理接口

本接口限企业铂金版客户专用，请前往铂金版购买页面查看详情。

### 描述

OnsConsumerAccumulate 接口用于查询指定订阅组的消费堆积情况，可以获取到订阅组当前未消费的消息数以及大概的延迟时间。

### 使用场景

消费堆积查询一般在生产环境中需要关注订阅组消费进度时使用，用于粗略判断消息消费情况和延迟情况。

### 请求参数列表

| 名称 | 类型 | 是否必须 | 描述 |
|----|----|------|----|
|----|----|------|----|

|              |         |   |                                      |
|--------------|---------|---|--------------------------------------|
| OnsRegionId  | String  | 是 | 当前查询 MQ 所在区域，可以通过 ONSRegionList 方法获取 |
| OnsPlatform  | String  | 否 | 该请求来源，默认是从 POP 平台                    |
| PreventCache | Long    | 是 | 用于 CSRF 校验，设置为系统当前时间即可               |
| ConsumerId   | String  | 是 | 需要查询的消费端 CID                         |
| Detail       | Boolean | 否 | 是否查询详细信息，默认为否                        |

## 返回参数列表

| 名称        | 类型     | 描述                  |
|-----------|--------|---------------------|
| RequestId | String | 为公共参数，每个请求独一无二      |
| HelpUrl   | String | 帮助链接                |
| Data      | Data   | 指定 Consumer 的消费堆积情况 |

## Data 数据结构

| 成员                | 类型                                                       | 描述                                 |
|-------------------|----------------------------------------------------------|------------------------------------|
| OnLine            | Boolean                                                  | 当前消费者是否在线                          |
| TotalDiff         | Long                                                     | 当前 Consumer ID 订阅的所有 Topic 的消息总堆积数 |
| ConsumeTps        | Float                                                    | 当前消费 TPS                           |
| LastTimestamp     | Long                                                     | 最后更新时刻                             |
| DelayTime         | Long                                                     | 延迟时间                               |
| DetailInTopicList | List(OnsConsumerAccumulateResponse.Data.DetailInTopicDo) | 各个 Topic 具体情况                      |

## DetailInTopicList 数据结构

| 成员            | 类型     | 描述               |
|---------------|--------|------------------|
| Topic         | String | Topic 名称         |
| TotalDiff     | Long   | 当前 Topic 的消费堆积   |
| LastTimestamp | Long   | 最后更新时刻           |
| DelayTime     | Long   | 当前 Topic 的消费延迟时间 |

## 相关 API

- OnsConsumerStatus：消费者状态详情查询
- OnsConsumerConnection：消费者连接查询
- OnsResetOffset：消费清理堆积

## 使用示例

```
public static void main(String []args) {
    String regionId = "cn-hangzhou";
    String accessKey = "XXXXXXXXXXXXXXXXXX";
    String secretKey = "XXXXXXXXXXXXXXXXXX";
    String endPointName = "cn-hangzhou";
    String productName = "Ons";
    String domain = "ons.cn-hangzhou.aliyuncs.com";

    /**
     *根据自己需要访问的区域选择Region，并设置对应的接入点
     */
    try {
        DefaultProfile.addEndpoint(endPointName, regionId, productName, domain);
    } catch (ClientException e) {
        e.printStackTrace();
    }
    IClientProfile profile= DefaultProfile.getProfile(regionId, accessKey, secretKey);
    IAcsClient iAcsClient= new DefaultAcsClient(profile);
    OnsConsumerAccumulateRequest request = new OnsConsumerAccumulateRequest();
    // request.setCluster("taobaodaily");
    /**
     *ONSRegionId是指你需要API访问MQ哪个区域的资源.
     *该值必须要根据OnsRegionList方法获取的列表来选择和配置，因为OnsRegionId是变动的，不能够写固定值
     */
    request.setOnsRegionId("daily");
    request.setPreventCache(System.currentTimeMillis());
    request.setAcceptFormat(FormatType.JSON);
    request.setDetail(true);
    request.setConsumerId("RTDSQ_1013_GROUP");
    try {
        OnsConsumerAccumulateResponse response=iAcsClient.getAcsResponse(request);
        OnsConsumerAccumulateResponse.Data data =response.getData();
        System.out.println(data.getOnline()+" "+data.getTotalDiff()+" "+data.getConsumeTps()+" "
            +data.getDelayTime()+" "+data.getLastTimestamp());
        for (OnsConsumerAccumulateResponse.Data.DetailInTopicDo detailInTopicDo:data.getDetailInTopicList()){
            System.out.println(detailInTopicDo.getTopic()+" "
                +detailInTopicDo.getTotalDiff()+" "
                +detailInTopicDo.getLastTimestamp()+" "+detailInTopicDo.getDelayTime());
        }
    } catch (ServerException e) {
        e.printStackTrace();
    } catch (ClientException e) {
        e.printStackTrace();
    }
}
```

本接口限企业铂金版客户专用，请前往铂金版购买页面查看详情。

## 描述

OnsConsumerStatus 接口可以查询指定订阅组的详细状态数据，包含订阅关系检查，消费 TPS 统计，负载均衡状态，消费端连接等。

## 使用场景

消费者状态查询接口用于定位订阅组的详细状态，一般用于在粗略判断消费堆积以及客户端在线情况后，需要排查消费异常的原因时调用。可以判断出指定消费组(CID)的订阅关系是否一致，负载均衡是否正常，以及在线客户端的 Jstack 信息等。

**注意：**消费者状态查询接口需要调用后端大量接口完成数据聚合，因此查询速度慢，不建议频繁调用。

## 请求参数列表

| 名称           | 类型      | 是否必须 | 描述                                   |
|--------------|---------|------|--------------------------------------|
| OnsRegionId  | String  | 是    | 当前查询 MQ 所在区域，可以通过 ONsRegionList 方法获取 |
| OnsPlatform  | String  | 否    | 该请求来源，默认是从 POP 平台                    |
| PreventCache | Long    | 是    | 用于 CSRF 校验，设置为系统当前时间即可               |
| ConsumerId   | String  | 是    | 需要查询的消费端 CID                         |
| Detail       | Boolean | 否    | 是否查询详细信息                             |
| NeedJstack   | Boolean | 否    | 是否打印 JStack 信息                       |

## 返回参数列表

| 名称        | 类型               | 描述             |
|-----------|------------------|----------------|
| RequestId | String           | 为公共参数，每个请求独一无二 |
| HelpUrl   | String           | 帮助链接           |
| Data      | ConsumerStatusDo | 查询结果           |

## ConsumerStatusDo 数据结构

| 成员        | 类型      | 描述       |
|-----------|---------|----------|
| Online    | Boolean | 是否在线     |
| TotalDiff | Long    | 集群总的消费堆积 |

|                            |                                |                                    |
|----------------------------|--------------------------------|------------------------------------|
| ConsumeTps                 | Long                           | 总消费 TPS                            |
| LastTimestamp              | Long                           | 最后更新时间                             |
| DelayTime                  | Long                           | 延迟时间                               |
| ConsumeModel               | Long                           | 消费模型                               |
| SubscriptionSame           | Boolean                        | 订阅关系是否一致                           |
| RebalanceOK                | Boolean                        | 客户端 Rebalance 是否正常                 |
| ConnectionSet              | List(ConnectionDo)             | 该集群当前在线客户端信息                       |
| DetailInTopicList          | List(DetailInTopicDo)          | 各个 Topic 的消费情况                     |
| ConsumerConnectionInfoList | List(ConsumerConnectionInfoDo) | 该集群在线客户端详细信息，包含 Jstack、消费 RT 时间等信息 |

#### ConnectionDo 数据结构

| 成员         | 类型     | 描述          |
|------------|--------|-------------|
| ClientId   | String | 消费实例的 ID    |
| ClientAddr | String | 该消费实例的地址和端口 |
| Language   | String | 消费端语言       |
| Version    | String | 消费端版本       |

#### DetailInTopicDo 数据结构

| 成员            | 类型     | 描述             |
|---------------|--------|----------------|
| Topic         | String | Topic 名称       |
| TotalDiff     | Long   | 该 Topic 消费总堆积数 |
| LastTimestamp | Long   | 最后更新时间         |
| DelayTime     | Long   | 延迟时间           |

#### ConsumerConnectionInfoDo 数据结构

| 成员           | 类型      | 描述           |
|--------------|---------|--------------|
| ClientId     | String  | 消费实例的 ID     |
| Connection   | String  | 连接信息         |
| Language     | String  | 客户端语言        |
| Version      | String  | 客户端版本号       |
| ConsumeModel | String  | 消费类型，集群和广播两种 |
| ConsumeType  | String  | 从何处消费        |
| ThreadCount  | Integer | 消费线程数        |

|                 |                             |             |
|-----------------|-----------------------------|-------------|
| StartTimeStamp  | Long                        | 开始时间        |
| LastTimeStamp   | Long                        | 最后更新时间      |
| SubscriptionSet | List(SubscriptionData)      | 订阅关系集合      |
| RunningDataList | List(ConsumerRunningDataDo) | 实时状态统计      |
| Jstack          | List(ThreadTrackDo)         | Jstack 堆栈信息 |

SubscriptionData 数据结构

| 成员         | 类型           | 描述                     |
|------------|--------------|------------------------|
| Topic      | String       | 订阅的 Topic 名称           |
| SubString  | String       | 订阅该 Topic 的子类别 Tag 表达式 |
| SubVersion | Long         | 订阅关系版本号，为自增 long 型     |
| TagsSet    | List(String) | 订阅的 Tag 集合             |

ConsumerRunningDataDo 数据结构

| 成员                 | 类型     | 描述                 |
|--------------------|--------|--------------------|
| ConsumerId         | String | 订阅方的 ConsumerId 名称 |
| Topic              | String | 订阅的 Topic 名称       |
| Rt                 | Float  | 消费 RT 时间，单位 ms     |
| OkTps              | Float  | 消费消息成功的 TPS 统计     |
| FailedTps          | Float  | 消费消息失败的 TPS 统计     |
| FailedCountPerHour | Long   | 每小时内消费失败的消息数统计     |

ThreadTrackDo 数据结构

| 成员        | 类型           | 描述             |
|-----------|--------------|----------------|
| Thread    | String       | 线程名称           |
| TrackList | List(String) | Jstack 堆栈信息字符串 |

相关 API

- OnsConsumerAccumulate：查询消费堆积
- OnsConsumerConnection：查询消费方客户端连接

使用示例

```

public static void main(String []args) {
    String regionId = "cn-hangzhou";
    String accessKey = "XXXXXXXXXXXXXXXXXX";
    String secretKey = "XXXXXXXXXXXXXXXXXX";
    String endPointName = "cn-hangzhou";
    String productName = "Ons";
    String domain = "ons.cn-hangzhou.aliyuncs.com";

    /**
     *根据自己需要访问的区域选择Region，并设置对应的接入点
     */
    try {
        DefaultProfile.addEndpoint(endPointName, regionId, productName, domain);
    } catch (ClientException e) {
        e.printStackTrace();
    }
    IClientProfile profile= DefaultProfile.getProfile(regionId, accessKey, secretKey);
    IAcsClient iAcsClient= new DefaultAcsClient(profile);
    OnsConsumerStatusRequest request = new OnsConsumerStatusRequest();
    /**
     *ONSRegionId是指你需要API访问MQ哪个区域的资源.
     *该值必须要根据OnsRegionList方法获取的列表来选择和配置，因为OnsRegionId是变动的，不能够写固定值
     */
    request.setOnsRegionId("daily");
    request.setPreventCache(System.currentTimeMillis());
    request.setAcceptFormat(FormatType.JSON);
    request.setConsumerId("CID_Mingduan");
    request.setDetail(true);
    request.setNeedJstack(false);
    try {
        OnsConsumerStatusResponse response=iAcsClient.getAcsResponse(request);
        OnsConsumerStatusResponse.Data data =response.getData();
        List<OnsConsumerStatusResponse.Data.ConnectionDo> connectionDoList = data.getConnectionSet();
        List<OnsConsumerStatusResponse.Data.DetailInTopicDo> detailInTopicDoList =data.getDetailInTopicList();
        List<OnsConsumerStatusResponse.Data.ConsumerConnectionInfoDo> consumerConnectionInfoDoList
        =data.getConsumerConnectionInfoList();
        System.out.print(data.getOnline()+" "+data.getTotalDiff()+" "+data.getConsumeTps()+" "+
        data.getLastTimestamp()+" "+data.getDelayTime()+" "+data.getConsumeModel()+
        " "+data.getSubscriptionSame()+" "+data.getRebalanceOK());
    } catch (ServerException e) {
        e.printStackTrace();
    } catch (ClientException e) {
        e.printStackTrace();
    }
}

```

OnsConsumerConnection 接口用于查询指定订阅组(CID)下当前客户端的连接情况。

## 使用场景

一般用于判断指定 CID 的消费者是否在线，并获取详细的客户端连接的列表。

请求参数列表

| 名称           | 类型     | 是否必须 | 描述                                  |
|--------------|--------|------|-------------------------------------|
| OnsRegionId  | String | 是    | 当前查询 MQ 所在区域，可以通过ONSRegionList 方法获取 |
| OnsPlatform  | String | 否    | 该请求来源，默认是从POP 平台                    |
| PreventCache | Long   | 是    | 用于 CSRF 校验，设置为系统当前时间即可              |
| ConsumerId   | String | 是    | 需要查询的消费端 CID                        |

返回参数列表

| 名称        | 类型                 | 描述                |
|-----------|--------------------|-------------------|
| RequestId | String             | 为公共参数，每个请求独一无二    |
| HelpUrl   | String             | 帮助链接              |
| Data      | List(ConnectionDo) | 指定 Consumer 的连接信息 |

ConnectionDo 数据结构

| 成员         | 类型     | 描述          |
|------------|--------|-------------|
| ClientId   | String | 消费实例的 ID    |
| ClientAddr | String | 该消费实例的地址和端口 |
| Language   | String | 消费端语言       |
| Version    | String | 消费端版本       |

相关 API

- OnsConsumerStatus：消费者状态查询
- OnsConsumerAccumulate：消费堆积查询

使用示例

```
public static void main(String []args) {
String regionId = "cn-hangzhou";
String accessKey = "XXXXXXXXXXXXXXXXXX";
String secretKey = "XXXXXXXXXXXXXXXXXX";
String endPointName = "cn-hangzhou";
String productName ="Ons";
```

```
String domain ="ons.cn-hangzhou.aliyuncs.com";

/**
 *根据自己需要访问的区域选择Region，并设置对应的接入点
 */
try {
    DefaultProfile.addEndpoint(endPointName , regionId , productName , domain);
} catch (ClientException e) {
    e.printStackTrace();
}
IClientProfile profile= DefaultProfile.getProfile(regionId , accessKey , secretKey);
IAcsClient iAcsClient= new DefaultAcsClient(profile);
OnsConsumerGetConnectionRequest request = new OnsConsumerGetConnectionRequest();
/**
 *ONSRegionId是指你需要API访问MQ哪个区域的资源.
 *该值必须要根据OnsRegionList方法获取的列表来选择和配置，因为OnsRegionId是变动的，不能够写固定值
 */
request.setOnsRegionId("daily");
request.setPreventCache(System.currentTimeMillis());
request.setAcceptFormat(FormatType.JSON);
request.setConsumerId("RTDSQ_1013_GROUP");
try {
    OnsConsumerGetConnectionResponse response=iAcsClient.getAcsResponse(request);
    List<OnsConsumerGetConnectionResponse.Data.ConnectionDo>
    connectionDoList=response.getData().getConnectionList();
    for(OnsConsumerGetConnectionResponse.Data.ConnectionDo connectionDo:connectionDoList){
        System.out.println(connectionDo.getClientId()+" "+
            connectionDo.getClientAddr()+" "+
            connectionDo.getLanguage()+" "+
            connectionDo.getVersion());
    }
} catch (ServerException e) {
    e.printStackTrace();
} catch (ClientException e) {
    e.printStackTrace();
}
}
```

本接口限企业铂金版客户专用，请前往铂金版购买页面查看详情。

## 描述

根据用户指定的订阅组，将当前的订阅组消费位点重置到指定时间戳。一般用于清理堆积消息，或者回溯消费。有两种方式，一种是清理所有消息，一种是清理消费进度到指定的时间。

## 请求参数

| 名称          | 类型     | 是否必须 | 描述                                  |
|-------------|--------|------|-------------------------------------|
| OnsRegionId | String | 是    | 当前查询 MQ 所在区域，可以通过ONSRegionList 方法获取 |

|                |         |       |                             |
|----------------|---------|-------|-----------------------------|
| OnsPlatform    | String  | 否     | 该请求来源，默认是从 POP 平台           |
| PreventCache   | Long    | 是     | 用于 CSRF 校验，设置为系统当前时间即可      |
| ConsumerId     | String  | 是     | 需要查询的消费端CID                 |
| Topic          | String  | 是     | 需要重置位点的 Topic名称             |
| Type           | Integer | 是     | 设置0代表清除所有消息，设置1代表清理到指定时间    |
| ResetTimestamp | Long    | false | 重置位点到指定的时戳，仅仅在 Type 为 1 时生效 |

返回参数

| 名称        | 类型     | 描述             |
|-----------|--------|----------------|
| RequestId | String | 为公共参数，每个请求独一无二 |
| HelpUrl   | String | 帮助链接           |

使用示例

```
public static void main(String []args) {
String regionId = "cn-hangzhou";
String accessKey = "XXXXXXXXXXXXXXXXXXXX";
String secretKey = "XXXXXXXXXXXXXXXXXXXX";
String endPointName ="cn-hangzhou";
String productName ="Ons";
String domain ="ons.cn-hangzhou.aliyuncs.com";

/**
*根据自己所在的区域选择Region后,设置对应的接入点
*/
try {
DefaultProfile.addEndpoint(endPointName,regionId,productName,domain);
} catch (ClientException e) {
e.printStackTrace();
}
IClientProfile profile= DefaultProfile.getProfile(regionId,accessKey,secretKey);
IAcsClient iAcsClient= new DefaultAcsClient(profile);
OnsConsumerResetOffsetRequest request = new OnsConsumerResetOffsetRequest();
/**
*ONSRegionId是指你需要API访问ONS哪个区域的资源。
*该值必须要根据OnsRegionList方法获取的列表来选择和配置，因为OnsRegionId是变动的，不能够写固定值
*/
request.setOnsRegionId("daily");
request.setPreventCache(System.currentTimeMillis());
}
```

```

request.setAcceptFormat(FormatType.JSON);
request.setConsumerId("CID_Mingduan");
request.setTopic("MingduanTest");
request.setType(1);
request.setResetTimestamp(System.currentTimeMillis());
try {
    OnsConsumerResetOffsetResponse response=iAcsClient.getAcsResponse(request);
    System.out.println(response.getRequestId());
} catch (ServerException e) {
    e.printStackTrace();
} catch (ClientException e) {
    e.printStackTrace();
}
}
}

```

OnsConsumerTimeSpan 接口用于查询当前订阅组订阅的 Topic 的最新消息时间戳以及消费的最新时间戳。

## 使用场景

使用本接口可以解决查询某个 Topic 当前服务器上存在的消息的最新时间以及最早时间，以及查询消费端最近消费的时间的需求。一般用于展示消费进度的概况，和 ConsumeAccumulate 接口配合使用。

## 请求参数列表

| 名称           | 类型     | 是否必须 | 描述                                |
|--------------|--------|------|-----------------------------------|
| OnsRegionId  | String | 是    | 设置所要查询的区域，具体值通过 OnsRegionList接口获取 |
| OnsPlatform  | String | 否    | 该请求来源，默认是从 POP 平台                 |
| PreventCache | Long   | 是    | 用于 CSRF 校验，设置为系统当前时间即可            |
| ConsumerId   | String | 是    | 需要查询的消费端 CID                      |
| Topic        | String | 是    | 需要查询的消费的 Topic                    |

## 返回参数列表

| 名称        | 类型     | 描述             |
|-----------|--------|----------------|
| RequestId | String | 为公共参数，每个请求独一无二 |
| HelpUrl   | String | 帮助链接           |
| Data      | Data   | 查询结果           |

## Data 数据结构

| 成员               | 类型     | 描述                        |
|------------------|--------|---------------------------|
| Topic            | String | 查询的 Topic 名称              |
| MinTimeStamp     | Long   | 整个 Topic 目前存储的最新的消息的时间    |
| MaxTimeStamp     | Long   | 整个 Topic 目前存储的最老的消息的时间    |
| ConsumeTimeStamp | Long   | 当前 group 消费该 Topic 的最新的时间 |

## 相关 API

- OnsConsumerAccumulate：查询消费堆积
- OnsConsumerStatus：查询消费端详情
- OnsConsumerResetOffset：重置位点和清除消息

## 使用示例

```
public static void main(String []args) {
    String regionId = "cn-hangzhou";
    String regionId = "cn-hangzhou";
    String accessKey = "XXXXXXXXXXXXXXXXXX";
    String secretKey = "XXXXXXXXXXXXXXXXXX";
    String endPointName = "cn-hangzhou";
    String productName = "Ons";
    String domain = "ons.cn-hangzhou.aliyuncs.com";

    /**
     *根据自己需要访问的区域选择Region，并设置对应的接入点
     */
    try {
        DefaultProfile.addEndpoint(endPointName, regionId, productName, domain);
    } catch (ClientException e) {
        e.printStackTrace();
    }
    IClientProfile profile= DefaultProfile.getProfile(regionId, accessKey, secretKey);
    IAcsClient iAcsClient= new DefaultAcsClient(profile);
    OnsConsumerTimeSpanRequest request = new OnsConsumerTimeSpanRequest();
    /**
     *ONSRegionId是指你需要API访问MQ哪个区域的资源。
     *该值必须要根据OnsRegionList方法获取的列表来选择和配置，因为OnsRegionId是变动的，不能够写固定值
     */
    request.setOnsRegionId("daily");
    request.setPreventCache(System.currentTimeMillis());
    request.setAcceptFormat(FormatType.JSON);
    request.setConsumerId("CID_Mingduan");
    request.setTopic("MingduanTest12");
    try {
```

```
OnsConsumerTimeSpanResponse response=iAcsClient.getAcsResponse(request);
OnsConsumerTimeSpanResponse.Data data =response.getData();
System.out.println(data.getTopic()+"\n"+
data.getConsumeTimeStamp()+"\n"+
data.getMaxTimeStamp()+"\n"+
data.getMinTimeStamp());
} catch (ServerException e) {
e.printStackTrace();
} catch (ClientException e) {
e.printStackTrace();
}
}
```

## 消息查询接口

本接口限企业铂金版客户专用，请前往铂金版购买页面查看详情。

### 描述

OnsMessageTrack 接口用于判断目标 MsgId 对应的消息是否已经消费过。

### 使用场景

MessageTrack 接口一般用于判断指定的消息是否曾经被消费过，查询条件为单条消息的 MsgId。需要注意的是 MessageTrack 的实现基于 MQ 的内部位点机制，在大部分场景下判断结果可信，但如果应用方进行过重置位点已经清除消息等操作，则判断结果可能有误。

### 请求参数列表

| 名称           | 类型     | 是否必须 | 描述                                |
|--------------|--------|------|-----------------------------------|
| OnsRegionId  | String | 是    | 设置所要查询的区域，具体值通过 OnsRegionList接口获取 |
| OnsPlatform  | String | 否    | 该请求来源，默认是从 POP 平台                 |
| PreventCache | Long   | 是    | 用于 CSRF 校验，设置为系统当前时间即可            |
| Topic        | String | 是    | 消息的 Topic                         |
| MsgId        | String | 是    | 消息 ID                             |

### 返回参数列表

| 名称 | 类型 | 描述 |
|----|----|----|
|----|----|----|

|           |                    |                |
|-----------|--------------------|----------------|
| RequestId | String             | 为公共参数，每个请求独一无二 |
| HelpUrl   | String             | 帮助链接           |
| Data      | List(MessageTrack) | 查询结果           |

MessageTrack 数据结构

| 名称            | 类型     | 描述     |
|---------------|--------|--------|
| ConsumerGroup | String | 消费集群名称 |
| TrackType     | String | 当前状态   |
| ExceptionDesc | String | 异常描述   |

相关 API

- OnsMessageGetByMsgId：根据 MsgId 精确查询消息

使用示例

```
public static void main(String []args) {
String regionId = "cn-hangzhou";
String accessKey = "XXXXXXXXXXXXXXXXXX";
String secretKey = "XXXXXXXXXXXXXXXXXX";
String endPointName = "cn-hangzhou";
String productName ="Ons";
String domain ="ons.cn-hangzhou.aliyuncs.com";

/**
*根据自己需要访问的区域选择Region，并设置对应的接入点
*/
try {
DefaultProfile.addEndpoint(endPointName , regionId , productName , domain);
} catch (ClientException e) {
e.printStackTrace();
}
IClientProfile profile= DefaultProfile.getProfile(regionId , accessKey , secretKey);
IAcsClient iAcsClient= new DefaultAcsClient(profile);
OnsMessageTraceRequest request = new OnsMessageTraceRequest();
/**
*ONSRegionId是指你需要API访问MQ哪个区域的资源.
*该值必须要根据OnsRegionList方法获取的列表来选择和配置，因为OnsRegionId是变动的，不能够写固定值
*/
request.setOnsRegionId("daily");
request.setPreventCache(System.currentTimeMillis());
request.setAcceptFormat(FormatType.JSON);
request.setTopic("MingduanTest");
request.setMsgId("0A91883700001F90000001BF4723CB3E");
try {
OnsMessageTraceResponse response=iAcsClient.getAcsResponse(request);
List<OnsMessageTraceResponse.MessageTrack> trackList =response.getData();
}
```

```

for (OnsMessageTraceResponse.MessageTrack track:trackList){
    System.out.println(track.getConsumerGroup()+" "+track.getTrackType()+" "+track.getExceptionDesc());
}
} catch (ServerException e) {
    e.printStackTrace();
} catch (ClientException e) {
    e.printStackTrace();
}
}
}

```

OnsMessageGetByMsgKey 接口通过传入 Topic 和 MsgKey 进行模糊查询，得到符合条件的消息的信息列表。

## 使用场景

根据 MsgKey 查询消息属于模糊查询，由于业务方的 key 可能不唯一，所以查询结果可能为多条。一般用在业务方无法获取 MsgId 的场景下，先进行模糊查询得到 msgId 列表，再调用 OnsMessageGetByMsgId 接口查询。

## 请求参数列表

| 名称           | 类型     | 是否必须 | 描述                                   |
|--------------|--------|------|--------------------------------------|
| OnsRegionId  | String | 是    | 当前查询 MQ 所在区域，可以通过 OnsRegionList 方法获取 |
| OnsPlatform  | String | 否    | 该请求来源，默认是从 POP 平台                    |
| PreventCache | Long   | 是    | 用于 CSRF 校验，设置为系统当前时间即可               |
| Topic        | String | 是    | 需要查询的消息的 Topic                       |
| Key          | String | 是    | 需要查询的消息的 MsgKey                      |

## 返回参数列表

| 名称        | 类型                     | 描述             |
|-----------|------------------------|----------------|
| RequestId | String                 | 为公共参数，每个请求独一无二 |
| HelpUrl   | String                 | 帮助链接           |
| Data      | List(OnsRestMessageDo) | 查询结果           |

### OnsRestMessageDo 数据结构

| 名称 | 类型 | 描述 |
|----|----|----|
|----|----|----|

|                |                       |                 |
|----------------|-----------------------|-----------------|
| Topic          | String                | 消息的 Topic       |
| Flag           | Integer               | null            |
| PropertyList   | List(MessageProperty) | 消息属性列表          |
| Body           | String                | 消息体             |
| StoreSize      | Integer               | 消息大小            |
| BornTimestamp  | Long                  | 生成时间戳           |
| BornHost       | String                | 生成该消息的客户端实例     |
| StoreTimestamp | Long                  | 被 Broker 存储的时间戳 |
| StoreHost      | String                | 存储该消息的服务器实例     |
| MsgId          | String                | 消息 ID           |
| BodyCRC        | Integer               | 消息体 CRC 校验值     |
| ReconsumeTimes | Integer               | 消息重试消费的次数       |

## 相关 API

- OnsMessageGetByMsgId：根据 MsgId 精确查询消息
- OnsMessageGetByTopic：根据 Topic 进行时间范围查询

## 使用示例

```
public static void main(String []args) {
    String regionId = "cn-hangzhou";
    String accessKey = "XXXXXXXXXXXXXXXXXX";
    String secretKey = "XXXXXXXXXXXXXXXXXX";
    String endPointName = "cn-hangzhou";
    String productName = "Ons";
    String domain = "ons.cn-hangzhou.aliyuncs.com";

    /**
     *根据自己需要访问的区域选择Region，并设置对应的接入点
     */
    try {
        DefaultProfile.addEndpoint(endPointName, regionId, productName, domain);
    } catch (ClientException e) {
        e.printStackTrace();
    }
    IClientProfile profile= DefaultProfile.getProfile(regionId, accessKey, secretKey);
    IAcsClient iAcsClient= new DefaultAcsClient(profile);
    OnsMessageGetByKeyRequest request = new OnsMessageGetByKeyRequest();
    /**
     *ONSRegionId是指你需要API访问MQ哪个区域的资源.
     *该值必须要根据OnsRegionList方法获取的列表来选择和配置，因为OnsRegionId是变动的，不能够写固定值
     */
    request.setOnsRegionId("daily");
}
```

```
request.setPreventCache(System.currentTimeMillis());
request.setAcceptFormat(FormatType.JSON);
request.setTopic("MingduanTest");
request.setKey("hello");
try {
    OnsMessageGetByKeyResponse response = iAcsClient.getAcsResponse(request);
    List<OnsMessageGetByKeyResponse.OnsRestMessageDo> onsRestMessageDoList=response.getData();
    for(OnsMessageGetByKeyResponse.OnsRestMessageDo onsRestMessageDo:onsRestMessageDoList){
        byte[] messageBody = Base64.decode(onsRestMessageDo.getBody());
        String message =new String (messageBody);
        System.out.println(onsRestMessageDo.getTopic()+" "+message+" "+
            onsRestMessageDo.getFlag()+" "+
            onsRestMessageDo.getBornHost()+" "+
            onsRestMessageDo.getStoreSize()+" "+
            onsRestMessageDo.getStoreHost()+" "+
            onsRestMessageDo.getStoreTimestamp()+" "+
            onsRestMessageDo.getReconsumeTimes());
    }
} catch (ServerException e) {
    e.printStackTrace();
} catch (ClientException e) {
    e.printStackTrace();
}
```

OnsMessageGetByMsgId 接口通过传入 MsgId 查询指定消息的信息，包括发送时间，存储服务器，消息的 Key 和 Tag 等熟悉，以及是否被消费过的判断。

## 使用场景

消息查询功能用于判断消息是否发送成功，发送时间以及目标服务器等信息。

根据 MsgId 查询消息属于精确查询，查询条件需要的 MsgId 由每次发送成功的 SendResult 中获取，因此业务方必须存储每次发送消息的结果。

## 请求参数列表

| 名称           | 类型     | 是否必须 | 描述                                |
|--------------|--------|------|-----------------------------------|
| OnsRegionId  | String | 是    | 设置所要查询的区域，具体值通过 OnsRegionList接口获取 |
| OnsPlatform  | String | 否    | 该请求来源，默认是从 POP 平台                 |
| PreventCache | Long   | 是    | 用于 CSRF 校验，设置为系统当前时间即可            |
| MsgId        | String | 是    | 需要查询的消息的 MsgId                    |
| Topic        | String | 是    | 需要查询的消息的 Topic                    |

返回参数列表

| 名称        | 类型     | 描述             |
|-----------|--------|----------------|
| RequestId | String | 为公共参数，每个请求独一无二 |
| HelpUrl   | String | 帮助链接           |
| Data      | Data   | 查询结果           |

Data 数据结构

| 名称             | 类型                    | 描述          |
|----------------|-----------------------|-------------|
| Topic          | String                | 消息的 Topic   |
| Flag           | Integer               | null        |
| PropertyList   | List(MessageProperty) | 消息属性列表      |
| Body           | String                | 消息体         |
| StoreSize      | Integer               | 消息大小        |
| BornTimestamp  | Long                  | 生成时间戳       |
| BornHost       | String                | 生成该消息的客户端实例 |
| StoreTimestamp | Long                  | 被服务端存储的时间戳  |
| StoreHost      | String                | 存储该消息的服务器实例 |
| MsgId          | String                | 消息 ID       |
| BodyCRC        | Integer               | 消息体 CRC 校验值 |
| ReconsumeTimes | Integer               | 消息重试消费的次数   |

相关 API

- OnsMessageGetByMsgKey：根据 MsgKey 模糊匹配查询消息
- OnsMessageGetByTopic：根据 Topic 和时间范围查询消息

使用示例

```
public static void main(String []args) {
    String regionId = "cn-hangzhou";
    String accessKey = "XXXXXXXXXXXXXXXXXX";
    String secretKey = "XXXXXXXXXXXXXXXXXX";
    String endPointName ="cn-hangzhou";
    String productName ="Ons";
    String domain ="ons.cn-hangzhou.aliyuncs.com";

    /**
    *根据自己需要访问的区域选择Region，并设置对应的接入点
    */
}
```

```
*/
try {
    DefaultProfile.addEndpoint(endPointName , regionId , productName , domain);
} catch (ClientException e) {
    e.printStackTrace();
}
IClientProfile profile= DefaultProfile.getProfile(regionId , accessKey , secretKey);
IAcsClient iAcsClient= new DefaultAcsClient(profile);
OnsMessageGetByMsgIdRequest request = new OnsMessageGetByMsgIdRequest();
/**
 *ONSRegionId是指你需要API访问MQ哪个区域的资源.
 *该值必须要根据OnsRegionList方法获取的列表来选择和配置，因为OnsRegionId是变动的，不能够写固定值
 */
request.setOnsRegionId("daily");
request.setPreventCache(System.currentTimeMillis());
request.setAcceptFormat(FormatType.JSON);
request.setMsgId("0ADA91A600002A9F000002ADDA8137E8");
request.setTopic("TopicTest");
try {
    OnsMessageGetByMsgIdResponse response = iAcsClient.getAcsResponse(request);
    OnsMessageGetByMsgIdResponse.Data data =response.getData();
    byte[] msgbody= Base64.decode(data.getBody());
    String message= new String(msgbody);
    System.out.println(data.getTopic()+" "+
        message+" "+
        data.getFlag()+" "+
        data.getBornHost()+" "+
        data.getStoreSize()+" "+
        data.getStoreHost()+" "+
        data.getStoreTimestamp()+" "+
        data.getReconsumeTimes());
} catch (ServerException e) {
    e.printStackTrace();
} catch (ClientException e) {
    e.printStackTrace();
}
}
```

OnsMessagePageQueryByTopic 接口通过传入 Topic 和时间段，分页查询指定时间段内存在的所有消息。

## 使用场景

根据 Topic 查询消息属于范围查询，查询结果会通过分页的方式展示给调用方，仅适用于没有任何搜索条件(MsgId 和 MsgKey)情况下的查询。

**注意：**根据 Topic 查询，需要尽可能缩短查询的时间区间，否则匹配的消息过多，将无法定位问题。

## 调用方式

- 首先，传入 Topic，起止时间，以及每页的大小，进行分页查询，如果有消息，默认返回第一页的消息以及总页数和查询 Task ID。
- 根据返回的结果中的查询 Task ID，进行后续取消息，取消息时传入 Task ID 以及当前页数。

## 请求参数列表

| 名称           | 类型      | 是否必须 | 描述                                           |
|--------------|---------|------|----------------------------------------------|
| OnsRegionId  | String  | 是    | 当前查询 MQ 所在区域，可以通过 OnsRegionList 方法获取         |
| OnsPlatform  | String  | 否    | 该请求来源，默认是从 POP 平台                            |
| PreventCache | Long    | 是    | 用于 CSRF 校验，设置为系统当前时间即可                       |
| Topic        | String  | 是    | 需要查询的消息的 Topic                               |
| BeginTime    | Long    | 是    | 查询范围的起始时间戳                                   |
| EndTime      | Long    | 是    | 查询范围的终止时间戳                                   |
| TaskId       | String  | 否    | 查询任务的 ID，首次查询不需要输入，后续取消息必须传入，根据第一次的返回结果取出该字段 |
| CurrentPage  | Integer | 是    | 当前取第几页消息，从 1 开始递增，取消息时不可超过最大页数               |
| PageSize     | Integer | 否    | 分页查询，每页最多显示消息数量，默认是 20，最小 5 条，最多 50 条        |

## 返回参数列表

| 名称         | 类型         | 描述             |
|------------|------------|----------------|
| RequestId  | String     | 为公共参数，每个请求独一无二 |
| HelpUrl    | String     | 帮助链接           |
| MsgFoundDo | MsgFoundDo | 查询结果           |

## MsgFoundDo 数据结构

| 名称           | 类型     | 描述                            |
|--------------|--------|-------------------------------|
| TaskId       | String | 第一次新建查询时，返回的 Task ID，用于后续取消息用 |
| MaxPageCount | Long   | 查询结果最大页数                      |
| CurrentPage  | Long   | 当前位于第几页                       |

|      |                        |                                      |
|------|------------------------|--------------------------------------|
| Data | List(OnsRestMessageDo) | 当前页的消息集合，该类型与 messageGetById 的返回结果一致 |
|------|------------------------|--------------------------------------|

## 相关 API

- OnsMessageGetByMsgId：根据 MsgId 精确查询消息
- OnsMessageGetByMsgKey：根据 MsgKey 模糊查询消息

## 使用示例

### 新建查询调用示例

```
public static void main(String []args) {
    String regionId = "cn-hangzhou";
    String accessKey = "XXXXXXXXXXXXXXXXXX";
    String secretKey = "XXXXXXXXXXXXXXXXXX";
    String endPointName = "cn-hangzhou";
    String productName = "Ons";
    String domain = "ons.cn-hangzhou.aliyuncs.com";

    /**
     *根据自己需要访问的区域选择Region，并设置对应的接入点
     */
    try {
        DefaultProfile.addEndpoint(endPointName, regionId, productName, domain);
    } catch (ClientException e) {
        e.printStackTrace();
    }
    IClientProfile profile= DefaultProfile.getProfile(regionId, accessKey, secretKey);
    IAcsClient iAcsClient= new DefaultAcsClient(profile);
    OnsMessagePageQueryByTopicRequest request = new OnsMessagePageQueryByTopicRequest();
    /**
     *ONSRegionId是指你需要API访问MQ哪个区域的资源.
     *该值必须要根据OnsRegionList方法获取的列表来选择和配置，因为OnsRegionId是变动的，不能够写固定值
     */
    request.setOnsRegionId("daily");
    request.setPreventCache(System.currentTimeMillis());
    request.setAcceptFormat(FormatType.JSON);
    request.setTopic("MingduanTest");
    request.setBeginTime(System.currentTimeMillis()-24*3600*1000);
    request.setEndTime(System.currentTimeMillis());
    request.setCurrentPage(1);
    request.setPageSize(20);
    try {
        OnsMessagePageQueryByTopicResponse response = iAcsClient.getAcsResponse(request);
    } catch (ServerException e) {
        e.printStackTrace();
    } catch (ClientException e) {
        e.printStackTrace();
    }
}
```

## 分页取消息调用示例

```
public static void main(String []args) {
    String regionId = "cn-hangzhou";
    String accessKey = "XXXXX";
    String secretKey = "XXXXX";
    IClientProfile profile = DefaultProfile.getProfile(regionId, accessKey, secretKey);
    IAcsClient iAcsClient = new DefaultAcsClient(profile);
    OnsMessagePageQueryByTopicRequest request = new OnsMessagePageQueryByTopicRequest();
    /**
     *OnsRegionId是指你需要API访问MQ哪个区域的资源.
     *该值必须要根据OnsRegionList方法获取的列表来选择和配置，因为OnsRegionId是变动的，不能够写固定值
     */
    request.setOnsRegionId("daily");
    request.setPreventCache(System.currentTimeMillis());
    request.setAcceptFormat(FormatType.JSON);
    request.setTopic("MingduanTest");
    request.setBeginTime(System.currentTimeMillis()-24*3600*1000);
    request.setEndTime(System.currentTimeMillis());
    request.setCurrentPage(3);
    request.setTaskId("0ADA91A600002A9F000002ADDA8137E8");
    try {
        OnsMessagePageQueryByTopicResponse response = iAcsClient.getAcsResponse(request);
    } catch (ServerException e) {
        e.printStackTrace();
    } catch (ClientException e) {
        e.printStackTrace();
    }
}
```

## 发布订阅统计接口

OnsTrendTopicInputTps 接可以查询一段时间内指定 Topic 的消息写入报表数据。

### 使用场景

线上环境统计 Topic 的消息数据可以使用该接口获取信息，根据类型可以选择查询指定时间段内 Topic 的写入量或者 TPS 曲线数据。

如果应用方消息量比较小，间隔不均匀，此时采用 TPS 查询可能会导致数据不明显，建议使用总量查询。

### 请求参数列表

| 名称          | 类型     | 是否必须 | 描述                |
|-------------|--------|------|-------------------|
| OnsRegionId | String | 是    | 当前查询 MQ 所在区域，可以通过 |

|              |         |   |                        |
|--------------|---------|---|------------------------|
|              |         |   | OnsRegionList 方法获取     |
| OnsPlatform  | String  | 否 | 该请求来源，默认是从 POP 平台      |
| PreventCache | Long    | 是 | 用于 CSRF 校验，设置为系统当前时间即可 |
| Topic        | String  | 是 | 需要查询的 Topic 名称         |
| BeginTime    | Long    | 是 | 查询区间起始毫秒时间戳            |
| EndTime      | Long    | 是 | 查询区间终止毫秒时间戳            |
| Period       | Long    | 是 | 采样周期，单位分钟，支持（1，5，10）   |
| Type         | Integer | 是 | 查询的类型（0代表总量，1代表TPS）    |

## 返回参数列表

| 名称        | 类型     | 描述             |
|-----------|--------|----------------|
| RequestId | String | 为公共参数，每个请求独一无二 |
| HelpUrl   | String | 帮助链接           |
| data      | Data   | 数据集            |

## Data 数据集定义

| 名称      | 类型                | 描述        |
|---------|-------------------|-----------|
| Title   | String            | table 的名称 |
| Records | List(StatsDataDo) | 采集点信息     |

## StatsDataDo 数据集定义

| 名称 | 类型    | 描述              |
|----|-------|-----------------|
| X  | Long  | 横轴，毫秒时间戳        |
| Y  | Float | 纵轴，数据（TPS 或者总量） |

## 相关 API

- OnsTrendGroupOutputTps：查询 CID 的消息消费统计

## 使用示例

```
public static void main(String []args) {
    String regionId = "cn-hangzhou";
    String accessKey = "XXXXXXXXXXXXXXXXXX";
    String secretKey = "XXXXXXXXXXXXXXXXXX";
    String endPointName = "cn-hangzhou";
    String productName = "Ons";
    String domain = "ons.cn-hangzhou.aliyuncs.com";

    /**
     *根据自己需要访问的区域选择Region，并设置对应的接入点
     */
    try {
        DefaultProfile.addEndpoint(endPointName, regionId, productName, domain);
    } catch (ClientException e) {
        e.printStackTrace();
    }
    IClientProfile profile= DefaultProfile.getProfile(regionId, accessKey, secretKey);
    IAcsClient iAcsClient= new DefaultAcsClient(profile);
    OnsTrendTopicInputTpsRequest request =new OnsTrendTopicInputTpsRequest();
    request.setOnsRegionId("cn-qingdao-publictest");
    request.setPreventCache(System.currentTimeMillis());
    request.setAcceptFormat(FormatType.JSON);
    request.setTopic("MingduanTestAPI");
    request.setBeginTime(System.currentTimeMillis()-4*3600*1000);
    request.setEndTime(System.currentTimeMillis());
    request.setPeriod(1);
    request.setType(0);
    try {
        OnsTrendTopicInputTpsResponse response =iAcsClient.getAcsResponse(request);
        OnsTrendTopicInputTpsResponse.Data data =response.getData();
        System.out.println(data.getTitle()+"\n"+
            data.getRecords());
    } catch (ClientException e) {
        e.printStackTrace();
    }
}
```

OnsTrendGroupOutputTps 接口可以查询指定 Consumer ID 在一段时间内获消费消息的统计信息。

## 使用场景

线上环境生成消费数据报表时，可以根据该接口查询消费消息的数量或者 TPS 统计。

### 查询场景:

- 查询消费总量曲线的统计
- 查询消费 TPS 曲线的统计

如果应用方消息量比较小，间隔不均匀，此时采用 TPS 查询可能会导致数据不明显，建议使用总量查询。

## 请求参数列表

| 名称           | 类型      | 是否必须 | 描述                                   |
|--------------|---------|------|--------------------------------------|
| OnsRegionId  | String  | 是    | 当前查询 MQ 所在区域，可以通过 OnsRegionList 方法获取 |
| OnsPlatform  | String  | 否    | 该请求来源，默认是从 POP 平台                    |
| PreventCache | Long    | 是    | 用于 CSRF 校验，设置为系统当前时间即可               |
| Topic        | String  | 是    | 需要查询的 Topic 名称                       |
| ConsumerId   | String  | 是    | 需要查询的 CID 名称                         |
| BeginTime    | Long    | 是    | 查询区间起始毫秒时间戳                          |
| EndTime      | Long    | 是    | 查询区间终止毫秒时间戳                          |
| Period       | Long    | 是    | 采样周期，单位分钟，支持（1，5，10）                 |
| Type         | Integer | 是    | 查询的类型（0代表总量，1代表TPS）                  |

## 返回参数列表

| 名称        | 类型     | 描述             |
|-----------|--------|----------------|
| RequestId | String | 为公共参数，每个请求独一无二 |
| HelpUrl   | String | 帮助链接           |
| data      | Data   | 数据集            |

## Data 数据集定义

| 名称      | 类型                | 描述        |
|---------|-------------------|-----------|
| Title   | String            | table 的名称 |
| Records | List(StatsDataDo) | 采集点信息     |

## StatsDataDo 数据集定义

| 名称 | 类型    | 描述              |
|----|-------|-----------------|
| X  | Long  | 横轴，毫秒时间戳        |
| Y  | Float | 纵轴，数据（TPS 或者总量） |

## 相关 API

- OnsTrendTopicInputTps：查询消息写入的统计信息

## 使用示例

```
public static void main(String []args) {
    String regionId = "cn-hangzhou";
    String accessKey = "XXXXXXXXXXXXXXXXXX";
    String secretKey = "XXXXXXXXXXXXXXXXXX";
    String endPointName = "cn-hangzhou";
    String productName = "Ons";
    String domain = "ons.cn-hangzhou.aliyuncs.com";

    /**
     *根据自己需要访问的区域选择Region，并设置对应的接入点
     */
    try {
        DefaultProfile.addEndpoint(endPointName, regionId, productName, domain);
    } catch (ClientException e) {
        e.printStackTrace();
    }
    IClientProfile profile= DefaultProfile.getProfile(regionId, accessKey, secretKey);
    IAcsClient iAcsClient= new DefaultAcsClient(profile);
    OnsTrendGroupOutputTpsRequest request =new OnsTrendGroupOutputTpsRequest();
    request.setOnsRegionId("cn-qingdao-publictest");
    request.setPreventCache(System.currentTimeMillis());
    request.setAcceptFormat(FormatType.JSON);
    request.setTopic("MingduanTestAPI");
    request.setConsumerId("CID_Mingduan");
    request.setBeginTime(System.currentTimeMillis()-4*3600*1000);
    request.setEndTime(System.currentTimeMillis());
    request.setPeriod(1);
    request.setType(0);
    try {
        OnsTrendGroupOutputTpsResponse response =iAcsClient.getAcsResponse(request);
        OnsTrendGroupOutputTpsResponse.Data data =response.getData();
        System.out.println(data.getTitle()+"\n"+
            data.getRecords());
    } catch (ClientException e) {
        e.printStackTrace();
    }
}
```

## 资源授权接口

OnsEmpowerCreate 接口可以将目标 Topic 授权给指定用户使用，包括发布消息和订阅消息的权限。

## 使用场景

MQ 的 Topic 资源属于申请的 Owner 账号所有，当需要跨账号访问 Topic 资源时，Owner 账号可以使用授权接口对目标账号进行发布消息或者接收消息的权限授予。

### 授权注意事项:

- 授权可以支持三种权限：发布、订阅、发布&订阅。
- 发起授权的账号必须是资源的 Owner 账号，目标账号可以是子账号或者主账号。
- 授权后，目标账号仍然需要创建自己的订阅 CID，而不是使用 Owner 账号的 CID 来收消息。

## 主子帐号约束

本接口仅限主帐号使用，RAM 子帐号无法操作。

## 请求参数列表

| 名称           | 类型      | 是否必须 | 描述                                   |
|--------------|---------|------|--------------------------------------|
| OnsRegionId  | String  | 是    | 当前查询 MQ 所在区域，可以通过 OnsRegionList 方法获取 |
| OnsPlatform  | String  | 否    | 该请求来源，默认是从 POP 平台                    |
| PreventCache | Long    | 是    | 用于 CSRF 校验，设置为系统当前时间即可               |
| EmpowerUser  | Long    | 是    | 授权的目标用户 User ID，支持子账号 ID             |
| Topic        | String  | 是    | 授权的目标 Topic，该 Topic 必须是当前用户所有        |
| Relation     | Integer | 是    | 授权类型，2代表授权发送，4代表授权订阅，6代表授权发送和订阅      |

## 返回参数列表

| 名称        | 类型     | 描述             |
|-----------|--------|----------------|
| RequestId | String | 为公共参数，每个请求独一无二 |
| HelpUrl   | String | 帮助链接           |

## 相关 API

- OnsEmpowerList：查看授权列表

- OnsEmpowerDelet：删除授权

## 使用示例

```
public static void main(String []args) {
public static void main(String []args) {
String regionId = "cn-hangzhou";
String accessKey = "XXXXXXXXXXXXXXXXXX";
String secretKey = "XXXXXXXXXXXXXXXXXX";
String endPointName = "cn-hangzhou";
String productName = "Ons";
String domain = "ons.cn-hangzhou.aliyuncs.com";

/**
*根据自己需要访问的区域选择Region，并设置对应的接入点
*/
try {
DefaultProfile.addEndpoint(endPointName, regionId, productName, domain);
} catch (ClientException e) {
e.printStackTrace();
}
IClientProfile profile= DefaultProfile.getProfile(regionId, accessKey, secretKey);
IAcsClient iAcsClient= new DefaultAcsClient(profile);
OnsEmpowerCreateRequest request = new OnsEmpowerCreateRequest();
request.setOnsRegionId("cn-qingdao-publictest");
request.setPreventCache(System.currentTimeMillis());
request.setAcceptFormat(FormatType.JSON);
request.setEmpowerUser("167535647680831312");
request.setTopic("topic_for_public_msg_test");
request.setRelation(2);
try {
OnsEmpowerCreateResponse response=iAcsClient.getAcsResponse(request);
System.out.println(response.getRequestId());
} catch (ServerException e) {
e.printStackTrace();
} catch (ClientException e) {
e.printStackTrace();
}
}
```

OnsEmpowerList 接口可以查询目标 Topic 对特定账号的授权信息。

## 使用场景

Topic 的所有者查询自己 owner 的 Topic 是否对特定账号授权，授权的类型可以通过该接口查询。

## 主子帐号约束

本接口仅限主帐号使用，RAM 子帐号无法操作。

请求参数列表

| 名称           | 类型     | 是否必须 | 描述                                   |
|--------------|--------|------|--------------------------------------|
| OnsRegionId  | String | 是    | 当前查询 MQ 所在区域，可以通过 OnsRegionList 方法获取 |
| OnsPlatform  | String | 否    | 该请求来源，默认是从 POP 平台                    |
| PreventCache | Long   | 是    | 用于 CSRF 校验，设置为系统当前时间即可               |
| EmpowerUser  | Long   | 是    | 查询的目标用户 User ID，支持子账号 ID             |
| Topic        | String | 否    | 查询的目标 Topic，该 Topic 必须是当前用户所有        |

返回参数列表

| 名称        | 类型                    | 描述             |
|-----------|-----------------------|----------------|
| RequestId | String                | 为公共参数，每个请求独一无二 |
| HelpUrl   | String                | 帮助链接           |
| Data      | List(AuthOwnerInfoDo) | 查询到的权限数据列表     |

AuthOwnerInfoDo 结构体内容

| 名称       | 类型      | 描述             |
|----------|---------|----------------|
| Topic    | String  | Topic 名称       |
| Owner    | Long    | 所有者 UserId     |
| Relation | Integer | 权限类型，参考授权的参数说明 |

相关 API

- OnsEmpowerCreate：创建授权信息
- OnsEmpowerDelete：删除授权信息

使用示例

```
public static void main(String []args) {
public static void main(String []args) {
String regionId = "cn-hangzhou";
String accessKey = "XXXXXXXXXXXXXXXXXX";
```

```
String secretKey = "XXXXXXXXXXXXXXXXXX";
String endPointName = "cn-hangzhou";
String productName = "Ons";
String domain = "ons.cn-hangzhou.aliyuncs.com";

/**
 *根据自己需要访问的区域选择Region，并设置对应的接入点
 */
try {
    DefaultProfile.addEndpoint(endPointName, regionId, productName, domain);
} catch (ClientException e) {
    e.printStackTrace();
}
IClientProfile profile = DefaultProfile.getProfile(regionId, accessKey, secretKey);
IAcsClient iAcsClient = new DefaultAcsClient(profile);
OnsEmpowerListRequest request = new OnsEmpowerListRequest();
request.setOnsRegionId("cn-qingdao-publictest");
request.setPreventCache(System.currentTimeMillis());
request.setAcceptFormat(FormatType.JSON);
request.setEmpowerUser("1675352326808602");
request.setTopic("topic_for_public_msg_test");
try {
    OnsEmpowerListResponse response = iAcsClient.getAcsResponse(request);
    System.out.println(response.getRequestId());
} catch (ServerException e) {
    e.printStackTrace();
} catch (ClientException e) {
    e.printStackTrace();
}
}
```

OnsEmpowerDelete 接口可以对已经创建的授权关系进行删除，以解除之前创建的跨账号访问资源的权限。

## 使用场景

在创建授权后，如果需要解除跨账号的 Topic 访问权限，Owner 账号可以调用该接口解除。

### 注意事项:

- 调用方必须是资源的 Owner 账号。
- 需要删除的授权信息必须存在。

## 主子帐号约束

本接口仅限主帐号使用，RAM 子帐号无法操作。

## 请求参数列表

| 名称          | 类型     | 是否必须 | 描述                                 |
|-------------|--------|------|------------------------------------|
| OnsRegionId | String | 是    | 当前查询 MQ 所在区域，可以通过 OnsRegionList 方法 |

|              |        |   |                               |
|--------------|--------|---|-------------------------------|
|              |        |   | 获取                            |
| OnsPlatform  | String | 否 | 该请求来源，默认是从 POP 平台             |
| PreventCache | Long   | 是 | 用于 CSRF 校验，设置为系统当前时间即可        |
| EmpowerUser  | Long   | 是 | 授权的目标用户 User ID，支持子账号 ID      |
| Topic        | String | 是 | 授权的目标 Topic，该 Topic 必须是当前用户所有 |

返回参数列表

| 名称        | 类型     | 描述             |
|-----------|--------|----------------|
| RequestId | String | 为公共参数，每个请求独一无二 |
| HelpUrl   | String | 帮助链接           |

相关 API

- OnsEmpowerCreate：创建授权接口
- OnsEmpowerList：查看授权信息接口

使用示例

```
public static void main(String []args) {
public static void main(String []args) {
String regionId = "cn-hangzhou";
String accessKey = "XXXXXXXXXXXXXXXXXX";
String secretKey = "XXXXXXXXXXXXXXXXXX";
String endPointName = "cn-hangzhou";
String productName ="Ons";
String domain ="ons.cn-hangzhou.aliyuncs.com";

/**
*根据自己需要访问的区域选择Region，并设置对应的接入点
*/
try {
DefaultProfile.addEndpoint(endPointName , regionId , productName , domain);
} catch (ClientException e) {
e.printStackTrace();
}
IClientProfile profile= DefaultProfile.getProfile(regionId , accessKey , secretKey);
IAcsClient iAcsClient= new DefaultAcsClient(profile);
OnsEmpowerDeleteRequest request = new OnsEmpowerDeleteRequest();
request.setOnsRegionId("cn-qingdao-publictest");
request.setPreventCache(System.currentTimeMillis());
```

```
request.setAcceptFormat(FormatType.JSON);
request.setEmpowerUser("16753563236808602");
request.setTopic("topic_for_public_msg_test");
try {
    OnsEmpowerDeleteResponse response=iAcsClient.getAcsResponse(request);
    System.out.println(response.getRequestId());
} catch (ServerException e) {
    e.printStackTrace();
} catch (ClientException e) {
    e.printStackTrace();
}
}
```

## LMQ 管理接口

OnsMqttGroupIdList 接口用于查询目标 Region 里当前用户所拥有的所有 GroupId。

### 使用场景

获取 GroupId 列表的接口一般用于管理用户所有 GroupId 的场景，首先获取列表，然后可以针对单个 GroupId 进行相关的数据查询。

### 请求参数列表

| 名称           | 类型     | 是否必须 | 描述                       |
|--------------|--------|------|--------------------------|
| OnsRegionId  | String | 是    | 当前操作的 MQ 所在区域，详情参见公共术语页面 |
| OnsPlatform  | String | 否    | 请求来源，默认是从 POP 平台         |
| PreventCache | Long   | 是    | 用于 CSRF 校验，设置为系统当前时间即可   |

### 返回参数列表

| 名称        | 类型            | 描述                      |
|-----------|---------------|-------------------------|
| RequestId | String        | 为公共参数，每个请求独一无二，用于排查定位问题 |
| HelpUrl   | String        | 帮助链接                    |
| Data      | MqttGroupIdDo | GroupId 信息数据结构          |

MqttGroupIdDo 数据结构

| 名称          | 类型      | 描述                         |
|-------------|---------|----------------------------|
| Id          | Long    | 数据编号                       |
| ChannelId   | Integer | 访问的途径编号，阿里云环境为 0           |
| OnsRegionId | String  | MQ 的 RegionId              |
| RegionName  | String  | MQ 的 Region 名称             |
| Owner       | String  | GroupId 所属账号               |
| GroupId     | String  | GroupId                    |
| Topic       | String  | 该 GroupId 关联的 Parent Topic |
| Status      | Integer | 当前状态（0 服务中 1 冻结 2 暂停）      |
| CreateTime  | Long    | 创建时间，单位是毫秒时间戳              |
| UpdateTime  | Long    | 最后更新时间，单位是毫秒时间戳            |

错误码列表

无

相关 API

- OnsMqttQueryClientByGroupId：根据 GroupId 查询当前在线客户端数量
- OnsMqttQueryHistoryOnline：根据 GroupId 查询历史在线数统计曲线

使用示例

本示例仅提供一个参考，从杭州接入点接入，获取杭州 Region 的所有 GroupId。

```
public static void main(String[] args) {
    String regionId = "cn-hangzhou";
    String accessKey = "XXXXXXXXXXXXXXXXXX";
    String secretKey = "XXXXXXXXXXXXXXXXXX";
    String endPointName = "cn-hangzhou";
    String productName = "Ons";
    String domain = "ons.cn-hangzhou.aliyuncs.com";

    /**
     *根据自己所在的区域选择Region后,设置对应的接入点
     */
    try {
        DefaultProfile.addEndpoint(endPointName,regionId,productName,domain);
```

```
    } catch (ClientException e) {
    e.printStackTrace();
    }
    IClientProfile profile= DefaultProfile.getProfile(regionId,accessKey,secretKey);
    IAcsClient iAcsClient= new DefaultAcsClient(profile);
    OnsMqttGroupIdListRequest request = new OnsMqttGroupIdListRequest();
    /**
    *OnsRegionId是指你需要API访问MQ哪个区域的资源。
    *该值必须要根据OnsRegionList方法获取的列表来选择和配置，因为OnsRegionId是变动的，不能够写固定值
    */
    request.setOnsRegionId("XXXX");
    request.setPreventCache(System.currentTimeMillis());
    request.setAcceptFormat(FormatType.JSON);
    try {
    OnsMqttGroupIdListResponse response=iAcsClient.getAcsResponse(request);
    List<OnsMqttGroupIdListResponse.MqttGroupIdDo> groupIdList=response.getData();
    for(OnsMqttGroupIdListResponse.MqttGroupIdDo groupDo:groupIdList){
    System.out.println(groupDo.getId()+" "+
    groupDo.getChannelId()+" "+
    groupDo.getOnsRegionId()+" "+
    groupDo.getRegionName()+" "+
    groupDo.getOwner()+" "+
    groupDo.getGroupId()+" "+
    groupDo.getTopic()+" "+
    groupDo.getStatus()+" "+
    groupDo.getCreateTime()+" "+
    groupDo.getUpdateTime());
    }
    } catch (ServerException e) {
    e.printStackTrace();
    } catch (ClientException e) {
    e.printStackTrace();
    }
    }
```

OnsMqttQueryClientByClientId 接口用于查询指定设备的在线信息以及订阅关系等数据，查询条件是 clientId。

使用场景

查询设备信息的接口一般用于线上追踪单个设备的运行状态及排查问题。输入 clientId 即可查到对应设备是否在线，设备地址，以及当前的订阅关系等信息。

请求参数列表

| 名称          | 类型     | 是否必须 | 描述                      |
|-------------|--------|------|-------------------------|
| OnsRegionId | String | 是    | 当前操作的 MQ 所在区域，详情参见公共术语。 |
| OnsPlatform | String | 否    | 请求来源，默认是从 POP 平台。       |

|              |        |   |                         |
|--------------|--------|---|-------------------------|
| PreventCache | Long   | 是 | 用于 CSRF 校验，设置为系统当前时间即可。 |
| ClientId     | String | 是 | 需要查询的目标 clientId。       |

返回参数列表

| 名称        | 类型               | 描述                       |
|-----------|------------------|--------------------------|
| RequestId | String           | 为公共参数，每个请求独一无二，用于排查定位问题。 |
| HelpUrl   | String           | 帮助链接                     |
| Data      | MqttClientInfoDo | 设备在线信息数据结构               |

MqttClientInfoDo 数据结构

| 名称              | 类型                   | 描述              |
|-----------------|----------------------|-----------------|
| Online          | Boolean              | 设备是否在线          |
| ClientId        | String               | 设备的 clientId 名称 |
| SocketChannel   | String               | 设备连接的 IP 地址     |
| LastTouch       | Long                 | 最后更新时间          |
| SubScriptonData | List(SubscriptionDo) | 该设备当前的订阅关系集合    |

SubscriptionDo 数据结构

| 名称          | 类型      | 描述                           |
|-------------|---------|------------------------------|
| ParentTopic | String  | MQTT 的一级父 Topic              |
| SubTopic    | String  | MQTT 的多级子 Topic，如果没有则为 NULL。 |
| Qos         | Integer | 订阅关系的 QoS 级别                 |

错误码列表

无

相关 API

OnsMqttQueryClientByTopic：根据 Topic 查询当前订阅该 Topic 的在线客户端数量。

使用示例

本示例仅提供一个参考，从杭州接入点接入，查询指定 clientId 的在线数据。

```
public static void main(String[] args) {
    String regionId = "cn-hangzhou";
    String accessKey = "XXXXXXXXXXXXXXXXXX";
    String secretKey = "XXXXXXXXXXXXXXXXXX";
    String endPointName = "cn-hangzhou";
    String productName = "Ons";
    String domain = "ons.cn-hangzhou.aliyuncs.com";

    /**
     * 根据自己所在的区域选择 Region 后,设置对应的接入点。
     */
    try {
        DefaultProfile.addEndpoint(endPointName,regionId,productName,domain);
    } catch (ClientException e) {
        e.printStackTrace();
    }
    IClientProfile profile= DefaultProfile.getProfile(regionId,accessKey,secretKey);
    IAcsClient iAcsClient= new DefaultAcsClient(profile);
    OnsMqttQueryClientByClientIdRequest request = new OnsMqttQueryClientByClientIdRequest();
    /**
     * ONsRegionId 是指你需要 API 访问 MQ 哪个区域的资源。
     * 该值必须要根据 OnsRegionList 方法获取的列表来选择和配置，因为 OnsRegionId 是变动的，不能够写固定值。
     */
    request.setOnsRegionId("XXXX");
    request.setPreventCache(System.currentTimeMillis());
    request.setAcceptFormat(FormatType.JSON);
    request.setClientId("GID_XXX@@@XXXXX");
    try {
        OnsMqttQueryClientByClientIdResponse response = iAcsClient.getAcsResponse(request);
        OnsMqttQueryClientByClientIdResponse.MqttClientInfoDo clientInfoDo = response.getMqttClientInfoDo();
        System.out.println(clientInfoDo.getOnline() + " " +
            clientInfoDo.getClientId() + " " +
            clientInfoDo.getLastTouch() + " " +
            clientInfoDo.getSocketChannel());
        for (OnsMqttQueryClientByClientIdResponse.MqttClientInfoDo.SubscriptionDo subscriptionDo :
            clientInfoDo.getSubScriptonData()) {
            System.out.println(subscriptionDo.getParentTopic() + " " + subscriptionDo.getSubTopic() + " " +
                subscriptionDo.getQos());
        }
    } catch (ServerException e) {
        e.printStackTrace();
    } catch (ClientException e) {
        e.printStackTrace();
    }
}
```

OnsMqttQueryClientByGroupId 接口根据 GroupId 统计属于该分组下的在线设备的数量。

## 使用场景

查询在线设备数量的接口一般用于业务分析，统计一个 GroupId 分组下终端设备的活跃程度。

请求参数列表

| 名称           | 类型     | 是否必须 | 描述                       |
|--------------|--------|------|--------------------------|
| OnsRegionId  | String | 是    | 当前操作的 MQ 所在区域，详情参见公共术语页面 |
| OnsPlatform  | String | 否    | 请求来源，默认是从 POP 平台         |
| PreventCache | Long   | 是    | 用于 CSRF 校验，设置为系统当前时间即可   |
| GroupId      | String | 是    | 需要查询的目标分组 GroupId        |

返回参数列表

| 名称              | 类型              | 描述                      |
|-----------------|-----------------|-------------------------|
| RequestId       | String          | 为公共参数，每个请求独一无二，用于排查定位问题 |
| HelpUrl         | String          | 帮助链接                    |
| MqttClientSetDo | MqttClientSetDo | 分组在线信息数据结构              |

MqttClientSetDo 数据结构

| 名称          | 类型   | 描述         |
|-------------|------|------------|
| OnlineCount | Long | 分组所有在线设备数量 |

错误码列表

无

相关 API

- OnsMqttQueryClientByTopic：根据 Topic 查询当前订阅该 Topic 的在线客户端数量

使用示例

本示例仅提供一个参考，从杭州接入点接入，查询指定 groupId 的在线数量信息。

```
public static void main(String[] args) {
    String regionId = "cn-hangzhou";
    String accessKey = "XXXXXXXXXXXXXXXXXX";
    String secretKey = "XXXXXXXXXXXXXXXXXX";
    String endPointName = "cn-hangzhou";
```

```
String productName ="Ons";
String domain ="ons.cn-hangzhou.aliyuncs.com";

/**
 *根据自己所在的区域选择Region后,设置对应的接入点
 */
try {
    DefaultProfile.addEndpoint(endPointName,regionId,productName,domain);
} catch (ClientException e) {
    e.printStackTrace();
}
IClientProfile profile= DefaultProfile.getProfile(regionId,accessKey,secretKey);
IAcsClient iAcsClient= new DefaultAcsClient(profile);
OnsMqttQueryClientByGroupIdRequest request = new OnsMqttQueryClientByGroupIdRequest();
/**
 *ONSRegionId是指你需要API访问MQ哪个区域的资源。
 *该值必须要根据OnsRegionList方法获取的列表来选择和配置，因为OnsRegionId是变动的，不能够写固定值
 */
request.setOnsRegionId("XXXX");
request.setPreventCache(System.currentTimeMillis());
request.setAcceptFormat(FormatType.JSON);
request.setGroupId("GID_XXXXX");
try {
    OnsMqttQueryClientByGroupIdResponse response = iAcsClient.getAcsResponse(request);
    OnsMqttQueryClientByGroupIdResponse.MqttClientSetDo clientSetDo = response.getMqttClientSetDo();
    System.out.println(clientSetDo.getOnlineCount() + " " +
        clientSetDo.getPersistCount() + " ");
} catch (ServerException e) {
    e.printStackTrace();
} catch (ClientException e) {
    e.printStackTrace();
}
}
```

OnsMqttQueryClientByTopic 接口根据 MQTT 的 Topic 统计订阅该 Topic 的在线设备的数量。

使用场景

根据 Topic 查询在线设备数量的接口一般用于业务分析。业务上订阅某个特定 Topic 的设备一般是参与某个特殊活动的设备，可以根据 Topic 查询参与该活动的设备数量。

请求参数列表

| 名称           | 类型     | 是否必须 | 描述                       |
|--------------|--------|------|--------------------------|
| OnsRegionId  | String | 是    | 当前操作的 MQ 所在区域，详情参见公共术语页面 |
| OnsPlatform  | String | 否    | 请求来源，默认是从 POP 平台         |
| PreventCache | Long   | 是    | 用于 CSRF 校验，设置为系统当前时间即可   |

|             |        |   |                        |
|-------------|--------|---|------------------------|
| ParentTopic | String | 是 | 需要查询的目标 MQTT 一级父 Topic |
| SubTopic    | String | 否 | 需要查询的目标子 Topic，如果没有则不填 |

返回参数列表

| 名称              | 类型              | 描述                      |
|-----------------|-----------------|-------------------------|
| RequestId       | String          | 为公共参数，每个请求独一无二，用于排查定位问题 |
| HelpUrl         | String          | 帮助链接                    |
| MqttClientSetDo | MqttClientSetDo | 分组在线信息数据结构              |

MqttClientSetDo 数据结构

| 名称          | 类型   | 描述         |
|-------------|------|------------|
| OnlineCount | Long | 分组所有在线设备数量 |

错误码列表

无

相关 API

- OnsMqttQueryClientByTopic：根据 Topic 查询当前订阅该 Topic 的在线客户端数量

使用示例

本示例仅提供一个参考，从杭州接入点接入，查询指定 groupId 的在线数量信息。

```
public static void main(String[] args) {
    String regionId = "cn-hangzhou";
    String accessKey = "XXXXXXXXXXXXXXXXXX";
    String secretKey = "XXXXXXXXXXXXXXXXXX";
    String endPointName = "cn-hangzhou";
    String productName = "Ons";
    String domain = "ons.cn-hangzhou.aliyuncs.com";

    /**
     *根据自己所在的区域选择Region后,设置对应的接入点
     */
    try {
        DefaultProfile.addEndpoint(endPointName,regionId,productName,domain);
    } catch (ClientException e) {
        e.printStackTrace();
    }
}
```

```
}
IClientProfile profile= DefaultProfile.getProfile(regionId,accessKey,secretKey);
IAcsClient iAcsClient= new DefaultAcsClient(profile);
OnsMqttQueryClientByTopicRequest request = new OnsMqttQueryClientByTopicRequest();
/**
 *ONSRegionId是指你需要API访问MQ哪个区域的资源。
 *该值必须要根据OnsRegionList方法获取的列表来选择和配置，因为OnsRegionId是变动的，不能够写固定值
 */
request.setOnsRegionId("XXXX");
request.setPreventCache(System.currentTimeMillis());
request.setAcceptFormat(FormatType.JSON);
request.setParentTopic("XXXX");
request.setSubTopic("/secTopic/0/1");//此处如果没有子topic则不填，如果有，一定是完整的信息
try {
    OnsMqttQueryClientByTopicResponse response = iAcsClient.getAcsResponse(request);
    OnsMqttQueryClientByTopicResponse.MqttClientSetDo clientSetDo = response.getMqttClientSetDo();
    System.out.println(clientSetDo.getOnlineCount() + " " +
        clientSetDo.getPersistCount() + " ");
} catch (ServerException e) {
    e.printStackTrace();
} catch (ClientException e) {
    e.printStackTrace();
}
}
```

OnsMqttQueryHistoryOnline 接口根据 GroupId 分组和给定的时间段查询历史在线设备的数量曲线。

使用场景

查询历史在线设备数曲线的功能一般用于生成业务报表。

请求参数列表

| 名称           | 类型     | 是否必须 | 描述                                |
|--------------|--------|------|-----------------------------------|
| OnsRegionId  | String | 是    | 当前操作的 MQ 所在区域，详情参见公共术语。           |
| OnsPlatform  | String | 否    | 请求来源，默认是从 POP 平台。                 |
| PreventCache | Long   | 是    | 用于 CSRF 校验，设置为系统当前时间即可。           |
| GroupId      | String | 是    | 需要查询的目标分组 GroupId。                |
| BeginTime    | Long   | 是    | 查询的起始时间                           |
| EndTime      | Long   | 是    | 查询的终止时间，起止时间范围建议尽可能在当天，否则后端会自动截断。 |

返回参数列表

| 名称        | 类型     | 描述              |
|-----------|--------|-----------------|
| RequestId | String | 为公共参数，每个请求独一无二。 |
| HelpUrl   | String | 帮助链接            |
| data      | Data   | 数据集合            |

Data 数据集定义

| 名称      | 类型                | 描述        |
|---------|-------------------|-----------|
| Title   | String            | table 的名称 |
| Records | List(StatsDataDo) | 采集点信息     |

StatsDataDo 数据集定义

| 名称 | 类型    | 描述        |
|----|-------|-----------|
| X  | Long  | 横轴，毫秒时间戳。 |
| Y  | Float | 纵轴，数量。    |

错误码列表

无

相关API

OnsMqttQueryClientByGroupId：根据 GroupId 查询当前在线客户端数量。

使用示例

本示例仅提供一个参考，从杭州接入点接入，查询指定 GroupId 过去一小时的在线数量信息。

```
public static void main(String[] args) {
    String regionId = "cn-hangzhou";
    String accessKey = "XXXXXXXXXXXXXXXXXX";
    String secretKey = "XXXXXXXXXXXXXXXXXX";
    String endPointName = "cn-hangzhou";
    String productName = "Ons";
    String domain = "ons.cn-hangzhou.aliyuncs.com";

    /**
     *根据自己所在的区域选择 Region 后,设置对应的接入点。
     */
    try {
```

```
DefaultProfile.addEndpoint(endPointName,regionId,productName,domain);
} catch (ClientException e) {
e.printStackTrace();
}
IClientProfile profile= DefaultProfile.getProfile(regionId,accessKey,secretKey);
IAcsClient iAcsClient= new DefaultAcsClient(profile);
OnsMqttQueryHistoryOnlineRequest request = new OnsMqttQueryHistoryOnlineRequest();
/**
*OnsRegionId 是指你需要 API 访问 MQ 哪个区域的资源。
*该值必须要根据 OnsRegionList 方法获取的列表来选择和配置，因为 OnsRegionId 是变动的，不能够写固定值。
*/
request.setOnsRegionId("XXXX");
request.setPreventCache(System.currentTimeMillis());
request.setAcceptFormat(FormatType.JSON);
request.setGroupId("XXX");
request.setBeginTime(System.currentTimeMillis()-1000*3600);
request.setEndTime(System.currentTimeMillis());
try {
OnsMqttQueryHistoryOnlineResponse response = iAcsClient.getAcsResponse(request);
OnsMqttQueryHistoryOnlineResponse.Data data =response.getData();
System.out.println(data.getTitle()+"\n"+
data.getRecords());
} catch (ServerException e) {
e.printStackTrace();
} catch (ClientException e) {
e.printStackTrace();
}
}
```

OnsMqttQueryMsgByPubInfo 接口根据发送端信息查询消息。

## 使用场景

根据发送方信息查询消息的接口一般用于线上排查问题，或者查询某条消息的推送到达率等场景使用。需要输入发送方的详细信息来定位特定的消息。条件越精确，匹配出的消息越精确。获取到消息的 TraceId 之后即可根据 TraceId 调用 OnsMqttQueryTraceByTraceId 接口查询该条消息的实际推送量。

## 请求参数列表

| 名称           | 类型     | 是否必须 | 描述                      |
|--------------|--------|------|-------------------------|
| OnsRegionId  | String | 是    | 当前操作的 MQ 所在区域，详情参见公共术语。 |
| OnsPlatform  | String | 否    | 请求来源，默认是从 POP 平台。       |
| PreventCache | Long   | 是    | 用于 CSRF 校验，设置为系统当前时间即可。 |
| Topic        | String | 是    | 需要查询的消息的完整 Topic。       |
| ClientId     | String | 是    | 该消息的发送方                 |

|           |      |   |                                        |
|-----------|------|---|----------------------------------------|
|           |      |   | ClientId。                              |
| BeginTime | Long | 是 | 消息发送时间的起始范围。                           |
| EndTime   | Long | 是 | 消息发送时间的终止范围，时间范围越小查询越精确，默认只允许查询30分钟区间。 |

返回参数列表

| 名称             | 类型                  | 描述                       |
|----------------|---------------------|--------------------------|
| RequestId      | String              | 为公共参数，每个请求独一无二,用于排查定位问题。 |
| HelpUrl        | String              | 帮助链接                     |
| MqttMessageDos | List(MqttMessageDo) | 匹配到的所有消息                 |

MqttMessageDo 数据结构

| 名称      | 类型     | 描述                             |
|---------|--------|--------------------------------|
| PubTime | Long   | 消息的精确发送时间                      |
| TraceId | String | 消息唯一的 TraceId，根据该 ID 可以查询到推送量。 |

错误码列表

无

相关 API

OnsMqttQueryTraceByTraceId：根据 TraceId 查询指定消息的推送量。

使用示例

本示例仅提供一个参考，从杭州接入点接入，查询指定信息查询符合条件的所有消息。

```
public static void main(String[] args) {
    String regionId = "cn-hangzhou";
    String accessKey = "XXXXXXXXXXXXXXXXXX";
    String secretKey = "XXXXXXXXXXXXXXXXXX";
    String endPointName = "cn-hangzhou";
    String productName = "Ons";
    String domain = "ons.cn-hangzhou.aliyuncs.com";

    /**
    *根据自己所在的区域选择 Region 后,设置对应的接入点。
    */
}
```

```

*/
try {
DefaultProfile.addEndpoint(endPointName,regionId,productName,domain);
} catch (ClientException e) {
e.printStackTrace();
}
IClientProfile profile= DefaultProfile.getProfile(regionId,accessKey,secretKey);
IAcsClient iAcsClient= new DefaultAcsClient(profile);
OnsMqttQueryMsgByPubInfoRequest request = new OnsMqttQueryMsgByPubInfoRequest();
/**
*ONSRegionId 是指你需要 API 访问 MQ 哪个区域的资源。
*该值必须要根据 OnsRegionList 方法获取的列表来选择和配置，因为 OnsRegionId 是变动的，不能够写固定值。
*/
request.setOnsRegionId("XXXX");
request.setPreventCache(System.currentTimeMillis());
request.setAcceptFormat(FormatType.JSON);
request.setBeginTime(System.currentTimeMillis()-1000*60);
request.setEndTime(System.currentTimeMillis());
request.setClientId("GID_XXX@@@XXXX");//发送方的 clientId
request.setTopic("XXX/XX/XXX");
try {
OnsMqttQueryMsgByPubInfoResponse response = iAcsClient.getAcsResponse(request);
List<OnsMqttQueryMsgByPubInfoResponse.MqttMessageDo> msgList = response.getMqttMessageDos();
for(OnsMqttQueryMsgByPubInfoResponse.MqttMessageDo msg:msgList){
System.out.println(msg.getpubTime() + " " +msg.getTraceId());
}
} catch (ServerException e) {
e.printStackTrace();
} catch (ClientException e) {
e.printStackTrace();
}
}
}

```

OnsMqttQueryMsgTransTrend 接口根据 Topic 等信息查询历史消息收发量。

## 使用场景

查询历史消息收发曲线功能一般用于生成数据报表，统计业务规模等场景。

## 请求参数列表

| 名称           | 类型     | 是否必须 | 描述                      |
|--------------|--------|------|-------------------------|
| OnsRegionId  | String | 是    | 当前操作的MQ所在区域，详情参见公共术语。   |
| OnsPlatform  | String | 否    | 请求来源，默认是从 POP 平台。       |
| PreventCache | Long   | 是    | 用于 CSRF 校验，设置为系统当前时间即可。 |
| ParentTopic  | String | 是    | 需要查询的一级父 Topic。         |

|           |         |   |                                           |
|-----------|---------|---|-------------------------------------------|
| SubTopic  | String  | 否 | 查询的子 Topic，如果没有或者查询所有的 Topic 信息，可以不填。     |
| MsgType   | String  | 否 | 消息类别，p2p 或者是 sub，如果不填则默认查所有消息，填其他值无效。     |
| TpsType   | String  | 是 | 查询类别，TPS 或者 SUM，分别代表是查询总量还是 TPS 信息，其他值无效。 |
| TransType | String  | 是 | 收发类别，PUB 或者是 SUB，代表查询发送还是接收，其他值无效。        |
| Qos       | Integer | 是 | 查询的 QoS 级别，0,1,2，填其他值或者不填都默认查所有消息。        |
| BeginTime | Long    | 是 | 查询的起始时间                                   |
| EndTime   | Long    | 是 | 查询的终止时间，起止时间范围建议尽可能在当天，否则后端会自动截断。         |

## 返回参数列表

| 名称        | 类型     | 描述              |
|-----------|--------|-----------------|
| RequestId | String | 为公共参数，每个请求独一无二。 |
| HelpUrl   | String | 帮助链接            |
| data      | Data   | 数据集             |

## Data 数据集定义

| 名称      | 类型                | 描述        |
|---------|-------------------|-----------|
| Title   | String            | table 的名称 |
| Records | List(StatsDataDo) | 采集点信息     |

## StatsDataDo 数据集定义

| 名称 | 类型    | 描述               |
|----|-------|------------------|
| X  | Long  | 横轴，毫秒时间戳。        |
| Y  | Float | 纵轴，数据（TPS 或者总量）。 |

## 错误码列表

无

## 相关 API

无

## 使用示例

本示例仅提供一个参考，从杭州接入点接入，查询指定 Topic 过去一小时的消息发送数量信息。

```
public static void main(String[] args) {
    String regionId = "cn-hangzhou";
    String accessKey = "XXXXXXXXXXXXXXXXXX";
    String secretKey = "XXXXXXXXXXXXXXXXXX";
    String endPointName = "cn-hangzhou";
    String productName = "Ons";
    String domain = "ons.cn-hangzhou.aliyuncs.com";

    /**
     * 根据自己所在的区域选择 Region 后,设置对应的接入点。
     */
    try {
        DefaultProfile.addEndpoint(endPointName, regionId, productName, domain);
    } catch (ClientException e) {
        e.printStackTrace();
    }
    IClientProfile profile = DefaultProfile.getProfile(regionId, accessKey, secretKey);
    IAcsClient iAcsClient = new DefaultAcsClient(profile);
    OnsMqttQueryMsgTransTrendRequest request = new OnsMqttQueryMsgTransTrendRequest();
    /**
     * ONSRegionId 是指你需要 API 访问 MQ 哪个区域的资源。
     * 该值必须要根据 OnsRegionList 方法获取的列表来选择和配置，因为 OnsRegionId 是变动的，不能够写固定值。
     */
    request.setOnsRegionId("XXXX");
    request.setPreventCache(System.currentTimeMillis());
    request.setAcceptFormat(FormatType.JSON);
    request.setParentTopic("MQTT_TOPIC_ONSMONITOR_BJ");
    //request.setMsgType("sub");
    request.setTpsType("SUM");
    request.setTransType("PUB");
    //request.setQos(1);
    request.setBeginTime(System.currentTimeMillis()-1000*3600);
    request.setEndTime(System.currentTimeMillis());
    try {
        OnsMqttQueryMsgTransTrendResponse response = iAcsClient.getAcsResponse(request);
        OnsMqttQueryMsgTransTrendResponse.Data data = response.getData();
        System.out.println(data.getTitle()+"\n"+
            data.getRecords());
    } catch (ServerException e) {
        e.printStackTrace();
    }
}
```

```
} catch (ClientException e) {
e.printStackTrace();
}
}
```

OnsMqttQueryTraceByTraceId 接口根据消息 TraceId 查询指定消息的推送量。

使用场景

根据消息 TraceId 查询消息推送量的接口一般用于线上排查问题，或者统计特定消息的推送到达率等运营数据。

请求参数列表

| 名称           | 类型     | 是否必须 | 描述                                            |
|--------------|--------|------|-----------------------------------------------|
| OnsRegionId  | String | 是    | 当前操作的 MQ 所在区域，详情参见公共术语。                       |
| OnsPlatform  | String | 否    | 请求来源，默认是从 POP 平台。                             |
| PreventCache | Long   | 是    | 用于 CSRF 校验，设置为系统当前时间即可。                       |
| Topic        | String | 是    | 需要查询的消息的完整 Topic。                             |
| TraceId      | String | 是    | 该消息的 TraceId，由 OnsMqttQueryMsgByPubInfo 接口查询。 |

返回参数列表

| 名称        | 类型     | 描述                       |
|-----------|--------|--------------------------|
| RequestId | String | 为公共参数，每个请求独一无二，用于排查定位问题。 |
| HelpUrl   | String | 帮助链接                     |
| PushCount | Long   | 消息的实际推送量                 |

错误码列表

无

相关 API

OnsMqttQueryMsgByPubInfo：根据发送方信息查询消息，获取消息的 TraceId。

## 使用示例

本示例仅提供一个参考，从杭州接入点接入，根据给定的 TraceId 查询推送量。

```
public static void main(String[] args) {
    String regionId = "cn-hangzhou";
    String accessKey = "XXXXXXXXXXXXXXXXXX";
    String secretKey = "XXXXXXXXXXXXXXXXXX";
    String endPointName = "cn-hangzhou";
    String productName = "Ons";
    String domain = "ons.cn-hangzhou.aliyuncs.com";

    /**
     * 根据自己所在的区域选择 Region 后,设置对应的接入点。
     */
    try {
        DefaultProfile.addEndpoint(endPointName,regionId,productName,domain);
    } catch (ClientException e) {
        e.printStackTrace();
    }
    IClientProfile profile= DefaultProfile.getProfile(regionId,accessKey,secretKey);
    IAcsClient iAcsClient= new DefaultAcsClient(profile);
    OnsMqttQueryTraceByTraceIdRequest request = new OnsMqttQueryTraceByTraceIdRequest();
    /**
     * ONSRegionId 是指你需要 API 访问 MQ 哪个区域的资源。
     * 该值必须要根据 OnsRegionList 方法获取的列表来选择和配置，因为 OnsRegionId 是变动的，不能够写固定值。
     */
    request.setOnsRegionId("XXXX");
    request.setPreventCache(System.currentTimeMillis());
    request.setAcceptFormat(FormatType.JSON);
    request.setTraceId("XXXXXXXXXXxXXX");
    request.setTopic("XXX/XX/XXX");
    try {
        OnsMqttQueryTraceByTraceIdResponse response = iAcsClient.getAcsResponse(request);
        System.out.println("pushCount:"+response.getPushCount());
    } catch (ServerException e) {
        e.printStackTrace();
    } catch (ClientException e) {
        e.printStackTrace();
    }
}
```

## 消息轨迹接口

本接口限企业铂金版客户专用，请前往铂金版购买页面查看详情。

OnsTraceGetResultTest 接口通过传入轨迹查询任务的任务 ID 获取之前的轨迹查询结果。

## 使用场景

业务方已经根据 Message ID 或者 Message Key 创建了轨迹查询任务，并得到了查询任务的 ID。若需要根据该任务 ID 获取对应的轨迹结果的明细，调用该接口即可取出结果。因为消息轨迹查询任务耗时较多，创建任务后立即根据该接口获取结果可能为空，若遇到此情况，建议等待一段时间再重试。

## 请求参数列表

| 名称           | 类型     | 是否必须 | 描述                                 |
|--------------|--------|------|------------------------------------|
| OnsRegionId  | String | 是    | 设置所要查询的区域，具体值通过 OnsRegionList 接口获取 |
| OnsPlatform  | String | 否    | 该请求来源，默认是从 POP 平台                  |
| PreventCache | Long   | 是    | 用于 CSRF 校验，设置为系统当前时间即可             |
| QueryId      | String | 是    | 需要提取轨迹查询结果的任务 ID                   |

## 返回参数列表

| 名称        | 类型        | 描述             |
|-----------|-----------|----------------|
| RequestId | String    | 为公共参数，每个请求独一无二 |
| HelpUrl   | String    | 帮助链接           |
| TraceData | TraceData | 该查询任务的结果明细     |

## TraceData 数据结构

| 名称      | 类型     | 描述                                                                  |
|---------|--------|---------------------------------------------------------------------|
| queryId | String | 查询任务 ID                                                             |
| userId  | String | 查询用户 ID                                                             |
| topic   | String | 该查询任务的 Topic                                                        |
| msgId   | String | 该查询任务对应的 Message ID                                                 |
| msgKey  | String | 该查询任务对应的 Message Key                                                |
| status  | String | 该查询任务的结果，取值有 "finish"，"working"，"removed"，分别代表 "查询完成"，"查询中"，"任务已删除" |

|            |                  |              |
|------------|------------------|--------------|
| createTime | Long             | 该查询任务的创建时间   |
| updateTime | Long             | 该查询任务的最后更新时间 |
| traceList  | List(TraceMapDo) | 该查询任务的匹配轨迹列表 |

### TraceMapDo数据结构

| 名称           | 类型             | 描述                                                                                                                                      |
|--------------|----------------|-----------------------------------------------------------------------------------------------------------------------------------------|
| pubTime      | Long           | 消息发送时间                                                                                                                                  |
| topic        | String         | 消息的 Topic                                                                                                                               |
| pubGroupName | String         | 发送方客户端配置的 Producer ID                                                                                                                   |
| msgId        | String         | 消息的 Message ID                                                                                                                          |
| tag          | String         | 消息的 Tag                                                                                                                                 |
| msgKey       | String         | 消息的 Key                                                                                                                                 |
| bornHost     | String         | 消息发送方的客户端地址                                                                                                                             |
| costTime     | Integer        | 发送耗时，单位毫秒                                                                                                                               |
| status       | String         | 发送状态，取值有<br>“SEND_SUCCESS”，“SEND_FAILED”，“SEND_ROLLBACK”，“SEND_UNKNOWN”，“SEND_DELAY”，分别代表“发送成功”，“发送失败”，“事务消息回滚”，“事务消息未提交”，“定时(延时)消息定时中” |
| subList      | List(SubMapDo) | 消息的消费轨迹列表                                                                                                                               |

### SubMapDo数据结构

| 名称           | 类型                    | 描述                         |
|--------------|-----------------------|----------------------------|
| subGroupName | String                | 消费方 Consumer ID            |
| successCount | Integer               | 该 Consumer ID 消费成功次数统计     |
| failCount    | Integer               | 该 Consumer ID 消费失败次数统计     |
| clientList   | List(SubClientInfoDo) | 该 Consumer ID 客户端消费记录的明细列表 |

### SubClientInfoDo数据结构

| 名称 | 类型 | 描述 |
|----|----|----|
|----|----|----|

|                |         |                                                                                                 |
|----------------|---------|-------------------------------------------------------------------------------------------------|
| subGroupName   | String  | 该客户端所属的 Consumer ID                                                                             |
| subTime        | Long    | 消费开始时间戳                                                                                         |
| clientHost     | String  | 消费客户端的客户端地址                                                                                     |
| reconsumeTimes | Integer | 本地消费的投递轮次                                                                                       |
| costTime       | Integer | 本次消费耗时，单位毫秒                                                                                     |
| status         | String  | 消费状态，取值有<br>"CONSUME_FAILED"，"CONSUME_SUCCESS"，"CONSUME_NOT_RETURN"，分别代表“消费失败”，“消费成功”，“消费未返回结果” |

## 相关 API

- OnsTraceQueryByMsgId：根据 Message ID 精确查询消息轨迹。
- OnsTraceQueryByMsgKey：根据 Message Key 模糊查询消息轨迹。

## 使用示例

```
public static void main(String []args) {
    String regionId = "cn-hangzhou";
    String accessKey = "XXXXXXXXXXXXXXXXXX";
    String secretKey = "XXXXXXXXXXXXXXXXXX";
    String endPointName = "cn-hangzhou";
    String productName = "Ons";
    String domain = "ons.cn-hangzhou.aliyuncs.com";

    /**
     *根据自己需要访问的区域选择Region，并设置对应的接入点
     */
    try {
        DefaultProfile.addEndpoint(endPointName, regionId, productName, domain);
    } catch (ClientException e) {
        e.printStackTrace();
    }
    IClientProfile profile= DefaultProfile.getProfile(regionId, accessKey, secretKey);
    IAcsClient iAcsClient= new DefaultAcsClient(profile);
    OnsTraceGetResultRequest request = new OnsTraceGetResultRequest();
    /**
     *ONSRegionId是指你需要API访问MQ哪个区域的资源.
     *该值必须要根据OnsRegionList方法获取的列表来选择和配置，因为OnsRegionId是变动的，不能够写固定值
     */
    request.setOnsRegionId("daily");
    request.setPreventCache(System.currentTimeMillis());
    request.setAcceptFormat(FormatType.JSON);
    request.setQueryId("XXXXX");
    try {
        OnsTraceGetResultResponse response = iAcsClient.getAcsResponse(request);
    }
```

```
} catch (ServerException e) {  
    e.printStackTrace();  
}  
} catch (ClientException e) {  
    e.printStackTrace();  
}  
}
```

本接口限企业铂金版客户专用，请前往铂金版购买页面查看详情。

OnsTraceQueryByMsgId 接口通过传入 Topic 和 Message ID 创建轨迹查询任务，得到该查询任务的任务 ID。

使用场景

业务方记录了某消息的 Message ID，并需要根据此 Message ID 查询该消息的投递轨迹信息，即可使用 OnsTraceQueryByMsgId 接口创建查询任务。该接口返回任务 ID 后，根据OnsTraceGetResult 接口传入任务 ID 即可获取查询结果。

请求参数列表

| 名称           | 类型     | 是否必须 | 描述                                   |
|--------------|--------|------|--------------------------------------|
| OnsRegionId  | String | 是    | 当前查询 MQ 所在区域，可以通过 OnsRegionList 方法获取 |
| OnsPlatform  | String | 否    | 该请求的来源，默认是从 POP 平台                   |
| PreventCache | Long   | 是    | 用于 CSRF 校验，设置为系统当前时间即可               |
| Topic        | String | 是    | 需要查询的消息的 Topic                       |
| MsgId        | String | 是    | 需要查询的消息的 Message ID                  |
| BeginTime    | Long   | 是    | 查询的时间范围的起始时间点                        |
| EndTime      | Long   | 是    | 查询的时间范围的结束时间点                        |

返回参数列表

| 名称        | 类型     | 描述              |
|-----------|--------|-----------------|
| RequestId | String | 为公共参数，每个请求独一无二  |
| HelpUrl   | String | 帮助链接            |
| QueryId   | String | 该查询任务的任务 ID，根据该 |

|  |  |                                   |
|--|--|-----------------------------------|
|  |  | ID 再调用 OnsTraceGetResult 接口获取详细结果 |
|--|--|-----------------------------------|

## 相关 API

- OnsTraceQueryByMsgKey：根据 Message Key 模糊查询消息轨迹。
- OnsTraceGetResult：根据查询任务 ID 获取轨迹信息。

## 使用示例

```
public static void main(String []args) {
    String regionId = "cn-hangzhou";
    String accessKey = "XXXXXXXXXXXXXXXXXX";
    String secretKey = "XXXXXXXXXXXXXXXXXX";
    String endPointName = "cn-hangzhou";
    String productName = "Ons";
    String domain = "ons.cn-hangzhou.aliyuncs.com";

    /**
     *根据自己需要访问的区域选择Region，并设置对应的接入点
     */
    try {
        DefaultProfile.addEndpoint(endPointName, regionId, productName, domain);
    } catch (ClientException e) {
        e.printStackTrace();
    }
    IClientProfile profile= DefaultProfile.getProfile(regionId, accessKey, secretKey);
    IAcsClient iAcsClient= new DefaultAcsClient(profile);
    OnsTraceQueryByMsgIdRequest request = new OnsTraceQueryByMsgIdRequest();
    /**
     *ONSRegionId是指你需要API访问MQ哪个区域的资源.
     *该值必须要根据OnsRegionList方法获取的列表来选择和配置，因为OnsRegionId是变动的，不能够写固定值
     */
    request.setOnsRegionId("daily");
    request.setPreventCache(System.currentTimeMillis());
    request.setAcceptFormat(FormatType.JSON);
    request.setTopic("XXXXX");
    request.setMsgId("XXXXX");
    request.setBeginTime(System.currentTimeMillis()-1000*3600*24);
    request.setEndTime(System.currentTimeMillis());
    try {
        OnsTraceQueryByMsgIdResponse response = iAcsClient.getAcsResponse(request);
        System.out.print(response.getQueryId());
    } catch (ServerException e) {
        e.printStackTrace();
    } catch (ClientException e) {
        e.printStackTrace();
    }
}
```

本接口限企业铂金版客户专用，请前往铂金版购买页面查看详情。

OnsTraceQueryByMsgKey 接口通过传入 Topic 和 Message Key 创建轨迹查询任务，得到该查询任务的任务 ID。

## 使用场景

业务方记录了消息的 Message Key，并需要根据此 Key 查询该消息的投递轨迹信息，即可使用 OnsTraceQueryByMsgKey 接口创建查询任务。该接口返回任务 ID 后，再根据 OnsTraceGetResult 接口传入任务 ID 即可获取查询结果。

## 请求参数列表

| 名称           | 类型     | 是否必须 | 描述                                   |
|--------------|--------|------|--------------------------------------|
| OnsRegionId  | String | 是    | 当前查询 MQ 所在区域，可以通过 OnsRegionList 方法获取 |
| OnsPlatform  | String | 否    | 该请求的来源，默认是从 POP 平台                   |
| PreventCache | Long   | 是    | 用于 CSRF 校验，设置为系统当前时间即可               |
| Topic        | String | 是    | 需要查询的消息的 Topic                       |
| MsgKey       | String | 是    | 需要查询的消息的 Message Key                 |
| BeginTime    | Long   | 是    | 查询的时间范围的起始时间点                        |
| EndTime      | Long   | 是    | 查询的时间范围的结束时间点                        |

## 返回参数列表

| 名称        | 类型     | 描述                                                |
|-----------|--------|---------------------------------------------------|
| RequestId | String | 为公共参数，每个请求独一无二                                    |
| HelpUrl   | String | 帮助链接                                              |
| QueryId   | String | 该查询任务的任务 ID，根据该 ID 再调用 OnsTraceGetResult 接口获取详细结果 |

## 相关 API

- OnsTraceQueryByMsgId：根据 Message ID 精确查询消息轨迹。
- OnsTraceGetResult：根据查询任务 ID 获取轨迹信息。

## 使用示例

```
public static void main(String []args) {
    String regionId = "cn-hangzhou";
    String accessKey = "XXXXXXXXXXXXXXXXXXXX";
    String secretKey = "XXXXXXXXXXXXXXXXXXXX";
    String endPointName = "cn-hangzhou";
    String productName = "Ons";
    String domain = "ons.cn-hangzhou.aliyuncs.com";

    /**
    *根据自己需要访问的区域选择Region，并设置对应的接入点
    */
    try {
        DefaultProfile.addEndpoint(endPointName, regionId, productName, domain);
    } catch (ClientException e) {
        e.printStackTrace();
    }
    IClientProfile profile= DefaultProfile.getProfile(regionId, accessKey, secretKey);
    IAcsClient iAcsClient= new DefaultAcsClient(profile);
    OnsTraceQueryByMsgKeyRequest request = new OnsTraceQueryByMsgKeyRequest();
    /**
    *ONSRegionId是指你需要API访问MQ哪个区域的资源。
    *该值必须要根据OnsRegionList方法获取的列表来选择和配置，因为OnsRegionId是变动的，不能够写固定值
    */
    request.setOnsRegionId("daily");
    request.setPreventCache(System.currentTimeMillis());
    request.setAcceptFormat(FormatType.JSON);
    request.setTopic("XXXXXX");
    request.setMsgKey("XXXXXX");
    request.setBeginTime(System.currentTimeMillis()-1000*3600*24);
    request.setEndTime(System.currentTimeMillis());
    try {
        OnsTraceQueryByMsgKeyResponse response = iAcsClient.getAcsResponse(request);
        System.out.print(response.getQueryId());
    } catch (ServerException e) {
        e.printStackTrace();
    } catch (ClientException e) {
        e.printStackTrace();
    }
}
```