

云专线

接口参考

文档版本 01
发布日期 2024-06-28



版权所有 © 华为数字能源技术有限公司 2024。保留一切权利。

非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

商标声明



HUAWEI和其他华为商标均为华为技术有限公司的商标。

本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为数字能源技术有限公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为数字能源技术有限公司对本文档内容不做任何明示或暗示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

目录

1 使用前必读.....	1
1.1 概述.....	1
1.2 调用说明.....	1
1.3 终端节点 (Endpoint)	1
1.4 约束与限制.....	1
1.5 基本概念.....	1
2 API 概览.....	3
3 如何调用 API.....	4
3.1 构造请求.....	4
3.2 认证鉴权.....	7
3.3 返回结果.....	9
4 云专线接口使用说明.....	11
5 API.....	12
5.1 物理连接.....	12
5.1.1 查询物理连接详情.....	12
5.1.2 更新物理连接信息.....	21
5.1.3 删除物理连接.....	32
5.1.4 查询物理连接列表.....	35
5.1.5 查询租户的托管专线列表.....	45
5.1.6 创建托管专线连接.....	53
5.1.7 查询租户的托管专线详情.....	60
5.1.8 更新托管专线连接.....	67
5.1.9 删除托管专线连接.....	74
5.2 虚拟网关.....	78
5.2.1 查询虚拟网关详情.....	78
5.2.2 更新虚拟网关信息.....	84
5.2.3 删除虚拟网关.....	90
5.2.4 查询虚拟网关列表.....	94
5.2.5 创建虚拟网关.....	100
5.3 虚拟接口.....	110
5.3.1 查询虚拟接口详情.....	110
5.3.2 更新虚拟接口.....	120

5.3.3 删除虚拟接口.....	131
5.3.4 查询虚拟接口列表.....	134
5.3.5 创建虚拟接口.....	146
5.3.6 更新虚拟接口对等体.....	159
5.3.7 删除虚拟接口对应的对等体.....	166
5.3.8 创建虚拟接口对等体.....	169
5.3.9 执行虚拟接口倒换测试.....	176
5.3.10 查询虚拟接口倒换测试记录列表.....	182
5.4 标签管理.....	188
5.4.1 查询项目标签.....	188
5.4.2 查询资源标签.....	192
5.4.3 添加资源标签.....	196
5.4.4 批量添加删除资源标签.....	201
5.4.5 删除资源标签.....	208
5.4.6 通过标签查询资源实例.....	212
5.5 配额管理.....	227
5.5.1 查询配额.....	227
6 公共参数.....	232
6.1 通用请求返回值.....	232
6.2 错误码.....	233
A 附录.....	237
A.1 获取项目 ID.....	237
B 修订记录.....	239

1 使用前必读

1.1 概述

欢迎使用云专线（Direct Connect）服务。云专线用于搭建用户本地数据中心与云上VPC之间高速、低时延、稳定安全的专属连接通道，充分利用云服务优势的同时，继续使用现有的IT设施，实现灵活一体，可伸缩的混合云计算环境。

您可以使用本文档提供的API对云专线的资源进行相关操作，如创建、查询、修改、删除等。支持的全部操作请参见[API概览](#)。

在调用云专线服务的API之前，请确保已经充分了解云专线服务的相关概念，详细信息请参见[产品介绍](#)。

1.2 调用说明

云专线服务提供了REST（Representational State Transfer）风格API，支持您通过HTTPS请求调用，调用方法请参见[如何调用API](#)。

1.3 终端节点（Endpoint）

终端节点（Endpoint）即调用API的[请求地址](#)，不同服务不同区域的终端节点不同，您可以从[地区和终端节点](#)中查询云专线的终端节点。

1.4 约束与限制

您能创建的云专线服务资源的数量与配额有关系，如果您想查看服务配额、扩大配额，具体请参见“[约束与限制](#)”。

更详细的限制请参见具体API的说明。

1.5 基本概念

- 账号

用户注册时的账号，账号对其所拥有的资源及云服务具有完全的访问权限，可以重置用户密码、分配用户权限等。由于账号是付费主体，为了确保账号安全，建议您不要直接使用账号进行日常管理工作，而是创建用户并使用用户进行日常管理工作。

- 用户

由账号在IAM中创建的用户，是云服务的使用人员，具有身份凭证（密码和访问密钥）。

通常在调用API的鉴权过程中，您需要用到账号、用户和密码等信息。

- 区域（Region）

从地理位置和网络时延维度划分，同一个Region内共享弹性计算、块存储、对象存储、VPC网络、弹性公网IP、镜像等公共服务。Region分为通用Region和专属Region，通用Region指面向公共租户提供通用云服务的Region；专属Region指只承载同一类业务或只面向特定租户提供业务服务的专用Region。

详情请参见[区域和可用区](#)。

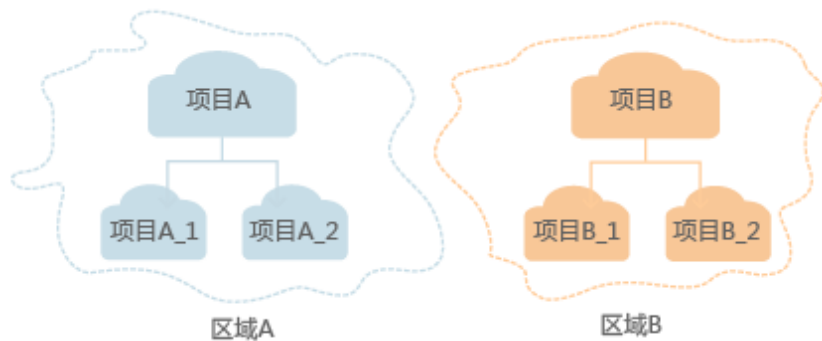
- 可用区（AZ，Availability Zone）

一个可用区是一个或多个物理数据中心的集合，有独立的风火水电，AZ内逻辑上再将计算、网络、存储等资源划分成多个集群。一个Region中的多个AZ间通过高速光纤相连，以满足用户跨AZ构建高可用性系统的需求。

- 项目

区域默认对应一个项目，这个项目由系统预置，用来隔离物理区域间的资源（计算资源、存储资源和网络资源），以默认项目为单位进行授权，用户可以访问您账号中该区域的所有资源。如果您希望进行更加精细的权限控制，可以在区域默认的项目中创建子项目，并在子项目中创建资源，然后以子项目为单位进行授权，使得用户仅能访问特定子项目中的资源，使得资源的权限控制更加精确。

图 1-1 项目隔离模型



- 企业项目

企业项目是项目的升级版，针对企业不同项目间的资源进行分组和管理，是逻辑隔离。企业项目中可以包含多个区域的资源，且项目中的资源可以迁入迁出。

关于企业项目ID的获取及企业项目特性的详细信息，请参见《[企业管理用户指南](#)》。

2 API 概览

云专线提供自研的REST接口。

通过使用云专线所提供的接口，您可以完整的使用云专线的所有功能。云专线的资源对象主要包括物理连接、虚拟网关、虚拟接口、标签管理等。

云专线提供的具体API如[表2-1](#)所示。

表 2-1 接口说明

子类型	说明
物理连接	对物理连接进行管理和操作，包括查询物理连接详细信息、更新物理连接信息、删除物理连接、查询物理连接列表、查询租户的托管专线列表、创建托管专线连接、查询租户的托管专线详情、更新托管专线连接以及删除托管专线连接等。
虚拟网关	对虚拟网关进行管理和操作，包括查询虚拟网关详情、修改虚拟网关信息、删除虚拟网关、查询虚拟网关列表以及创建虚拟网关等。
虚拟接口	对虚拟接口进行管理和操作，包括查询虚拟接口详情、修改虚拟接口、删除虚拟接口、查询虚拟接口列表以及创建虚拟接口等。
标签管理	对标签进行管理和操作，包括查询项目标签、查询资源标签、添加资源标签、批量添加删除资源标签、删除资源标签以及通过标签查询资源实例等。
配额管理	查询租户配额。

3 如何调用 API

3.1 构造请求

本节介绍REST API请求的组成，并以调用IAM服务的[管理员创建IAM用户](#)来说明如何调用API，该API获取用户的Token，Token可以用于调用其他API时鉴权。

请求 URI

请求URI由如下部分组成：

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

尽管请求URI包含在请求消息头中，但大多数语言或框架都要求您从请求消息中单独传递它，所以在此单独强调。

表 3-1 URI 中的参数说明

参数	描述
URI-scheme	表示用于传输请求的协议，当前所有API均采用HTTPS协议。
Endpoint	指定承载REST服务端点的服务器域名或IP，不同服务不同区域的Endpoint不同，您可以从 地区和终端节点 获取。 例如IAM服务在“中国-香港”区域的Endpoint为“iam.ap-southeast-1.myhuaweicloud.com”。
resource-path	资源路径，即API访问路径。从具体API的URI模块获取，例如“获取用户Token”API的resource-path为“/v3/auth/tokens”。
query-string	查询参数，是可选部分，并不是每个API都有查询参数。查询参数前面需要带一个“？”，形式为“参数名=参数取值”，例如“？limit=10”，表示查询不超过10条数据。

例如您需要获取IAM在“中国-香港”区域的Token，则需使用“中国-香港”区域的Endpoint（iam.ap-southeast-1.myhuaweicloud.com），并在[管理员创建IAM用户](#)的URI部分找到resource-path（/v3.0/OS-USER/users），拼接起来如下所示。


```
https://iam.ap-southeast-1.myhuaweicloud.com/v3.0/OS-USER/users
```

图 3-1 URI 示意图



说明

为方便查看，在每个具体API的URI部分，只给出resource-path部分，并将请求方法写在一起。这是因为URI-scheme都是HTTPS，而Endpoint在同一个区域也相同，所以简洁起见将这两部分省略。

请求方法

HTTP请求方法（也称为操作或动词），它告诉服务您正在请求什么类型的操作。

表 3-2 HTTP 方法

方法	说明
GET	请求服务器返回指定资源。
PUT	请求服务器更新指定资源。
POST	请求服务器新增资源或执行特殊操作。
DELETE	请求服务器删除指定资源，如删除对象等。
HEAD	请求服务器资源头部。
PATCH	请求服务器更新资源的部分内容。 当资源不存在的时候，PATCH可能会去创建一个新的资源。

在[管理员创建IAM用户](#)的URI部分，您可以看到其请求方法为“POST”，则其请求为：

```
POST https://iam.ap-southeast-1.myhuaweicloud.com/v3.0/OS-USER/users
```

请求消息头

附加请求头字段，如指定的URI和HTTP方法所要求的字段。例如定义消息体类型的请求头“Content-Type”，请求鉴权信息等。

详细的公共请求消息头字段请参见[表3-3](#)。

表 3-3 公共请求消息头

名称	描述	是否必选	示例
Host	请求的服务器信息，从服务API的URL中获取。值为hostname[:port]。端口缺省时使用默认的端口，https的默认端口为443。	否 使用AK/SK认证时该字段必选。	code.test.com or code.test.com:443
Content-Type	消息体的类型（格式）。推荐用户使用默认值application/json，有其他取值时会在具体接口中专门说明。	是	application/json
Content-Length	请求body长度，单位为Byte。	否	3495
X-Project-Id	project id，项目编号。请参考 获取项目ID 章节获取项目编号。	否 如果是专属云场景采用AK/SK认证方式的接口请求，或者多project场景采用AK/SK认证的接口请求，则该字段必选。	e9993fc787d94b6c886cb aa340f9c0f4
X-Auth-Token	用户Token。 用户Token也就是调用 获取用户Token 接口的响应值，该接口是唯一不需要认证的接口。 请求响应成功后在响应消息头（Headers）中包含的“X-Subject-Token”的值即为Token值。	否 使用Token认证时该字段必选。	注：以下仅为Token示例片段。 MIIPAgYJKoZlhvcNAQcCo ...ggg1BBIIlNPXsidG9rZ

📖 说明

API同时支持使用AK/SK认证，AK/SK认证使用SDK对请求进行签名，签名过程会自动往请求中添加Authorization（签名认证信息）和X-Sdk-Date（请求发送的时间）请求头。

AK/SK认证的详细说明请参见[认证鉴权](#)的“AK/SK认证”。

对于[管理员创建IAM用户](#)接口，使用AK/SK方式认证时，添加消息头后的请求如下所示。

```
POST https://iam.ap-southeast-1.myhuaweicloud.com/v3.0/OS-USER/users
Content-Type: application/json
X-Sdk-Date: 20240416T095341Z
Authorization: SDK-HMAC-SHA256 Access=*****, SignedHeaders=content-type;host;x-sdk-date,
Signature=*****
```

请求消息体（可选）

该部分可选。请求消息体通常以结构化格式（如JSON或XML）发出，与请求消息头中Content-Type对应，传递除请求消息头之外的内容。若请求消息体中的参数支持中文，则中文字符必须为UTF-8编码。

每个接口的请求消息体内容不同，也并不是每个接口都需要有请求消息体（或者说消息体为空），GET、DELETE操作类型的接口就不需要消息体，消息体具体内容需要根据具体接口而定。

对于[管理员创建IAM用户](#)接口，您可以从接口的请求部分看到所需的请求参数及参数说明，将消息体加入后的请求如下所示，其中加粗的字段需要根据实际值填写。

- **accountid**为IAM用户所属的账号ID。
- **username**为要创建的IAM用户名。
- **email**为IAM用户的邮箱。
- *********为IAM用户的登录密码。

```
POST https://iam.ap-southeast-1.myhuaweicloud.com/v3.0/OS-USER/users
Content-Type: application/json
X-Sdk-Date: 20240416T095341Z
Authorization: SDK-HMAC-SHA256 Access=*****, SignedHeaders=content-type;host;x-sdk-date,
Signature=*****

{
  "user": {
    "domain_id": "accountid",
    "name": "username",
    "password": "*****",
    "email": "email",
    "description": "IAM User Description"
  }
}
```

到这里为止这个请求需要的内容就具备齐全了，您可以使用[curl](#)、[Postman](#)或直接编写代码等方式发送请求调用API。对于获取用户Token接口，返回的响应消息头中的“X-Subject-Token”就是需要获取的用户Token。有了Token之后，您就可以使用Token认证调用其他API。

3.2 认证鉴权

调用接口有如下两种认证方式，您可以选择其中一种进行认证鉴权。

- AK/SK认证：通过AK（Access Key ID）/SK（Secret Access Key）加密调用请求。推荐使用AK/SK认证，其安全性比Token认证要高。

- Token认证：通过Token认证调用请求。

AK/SK 认证

📖 说明

AK/SK签名认证方式仅支持消息体大小在12MB以内，12MB以上的请求请使用Token认证。

AK/SK认证就是使用AK/SK对请求进行签名，在请求时将签名信息添加到消息头，从而通过身份认证。

- AK (Access Key ID)：访问密钥ID。与私有访问密钥关联的唯一标识符；访问密钥ID和私有访问密钥一起使用，对请求进行加密签名。
- SK (Secret Access Key)：私有访问密钥。与访问密钥ID结合使用，对请求进行加密签名，可标识发送方，并防止请求被修改。

使用AK/SK认证时，您可以基于签名算法使用AK/SK对请求进行签名，也可以使用专门的签名SDK对请求进行签名。详细的签名方法和SDK使用方法请参见[API签名指南](#)。

📖 说明

签名SDK只提供签名功能，与服务提供的SDK不同，使用时请注意。

Token 认证

📖 说明

Token的有效期为24小时，需要使用一个Token鉴权时，可以先缓存起来，避免频繁调用。

Token在计算机系统中代表令牌（临时）的意思，拥有Token就代表拥有某种权限。Token认证就是在调用API的时候将Token加到请求消息头中，从而通过身份认证，获得操作API的权限。Token可通过调用[获取用户Token](#)接口获取。

调用本服务API需要项目级别的Token，即调用[获取用户Token](#)接口时，请求body中auth.scope的取值需要选择project，如下所示。

```
{
  "auth": {
    "identity": {
      "methods": [
        "password"
      ],
      "password": {
        "user": {
          "name": "username", //IAM用户名
          "password": $ADMIN_PASS, //IAM用户密码，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全
          "domain": {
            "name": "domainname" //IAM用户所属账号名
          }
        }
      }
    },
    "scope": {
      "project": {
        "name": "xxxxxxx" //项目名称
      }
    }
  }
}
```

获取Token后，再调用其他接口时，您需要在请求消息头中添加“X-Auth-Token”，其值即为Token。例如Token值为“ABCDEFJ...”，则调用接口时将“X-Auth-Token: ABCDEFJ...”加到请求消息头即可，如下所示。

```
POST https://iam.ap-southeast-1.myhuaweicloud.com/v3/auth/projects
Content-Type: application/json
X-Auth-Token: ABCDEFJ....
```

3.3 返回结果

状态码

请求发送以后，您会收到响应，其中包含状态码、响应消息头和消息体。

状态码是一组从1xx到5xx的数字代码，状态码表示了请求响应的状态，完整的状态码列表请参见[通用请求返回值](#)。

对于[管理员创建IAM用户](#)接口，如果调用后返回状态码为“201”，则表示请求成功。

响应消息头

对应请求消息头，响应同样也有消息头，如“Content-type”。

对于[管理员创建IAM用户](#)接口，返回如[图3-2](#)所示的消息头，其中“X-Subject-Token”就是需要获取的用户Token。有了Token之后，您就可以使用Token认证调用其他API。

说明

建议在配置文件或者环境变量中密文存放，使用时解密，确保安全。

图 3-2 管理员创建 IAM 用户响应消息头

```
"X-Frame-Options": "SAMEORIGIN",
"X-IAM-ETag-id": "2562365939-d8f6f12921974cb097338ac11fceac8a",
"Transfer-Encoding": "chunked",
"Strict-Transport-Security": "max-age=31536000; includeSubdomains;",
"Server": "api-gateway",
"X-Request-Id": "af2953f2bcc67a42325a69a19e6c32a2",
"X-Content-Type-Options": "nosniff",
"Connection": "keep-alive",
"X-Download-Options": "noopen",
"X-XSS-Protection": "1; mode=block;",
"X-IAM-Trace-Id": "token_█_null_af2953f2bcc67a42325a69a19e6c32a2",
"Date": "Tue, 21 May 2024 09:03:40 GMT",
"Content-Type": "application/json; charset=utf8"
```

响应消息体（可选）

该部分可选。响应消息体通常以结构化格式（如JSON或XML）返回，与响应消息头中Content-Type对应，传递除响应消息头之外的内容。

对于[管理员创建IAM用户](#)接口，返回如下消息体。为篇幅起见，这里只展示部分内容。

```
{
  "user": {
```

```
"id": "c131886aec...",
"name": "IAMUser",
"description": "IAM User Description",
"areacode": "",
"phone": "",
"email": "***@**.com",
"status": null,
"enabled": true,
"pwd_status": false,
"access_mode": "default",
"is_domain_owner": false,
"xuser_id": "",
"xuser_type": "",
"password_expires_at": null,
"create_time": "2024-05-21T09:03:41.000000",
"domain_id": "d78cbac1.....",
"xdomain_id": "30086000.....",
"xdomain_type": "",
"default_project_id": null
}
}
```

当接口调用出错时，会返回错误码及错误信息说明，错误响应的Body体格式如下所示。

```
{
  "error_msg": "The request message format is invalid.",
  "error_code": "IMG.0001"
}
```

其中，error_code表示错误码，error_msg表示错误描述信息。

4 云专线接口使用说明

云专线有独立的Endpoint，接口调用时采用的是云专线（DCAAS）的API接口，调用时请注意使用对应服务的Endpoint进行调用。

5 API

5.1 物理连接

5.1.1 查询物理连接详情

功能介绍

查询物理连接详细信息。

调用方法

请参见[如何调用API](#)。

URI

GET /v3/{project_id}/dcaas/direct-connects/{direct_connect_id}

表 5-1 路径参数

参数	是否必选	参数类型	描述
direct_connect_id	是	String	物理专线连接ID。 最小长度：36 最大长度：36
project_id	是	String	租户项目ID

表 5-2 Query 参数

参数	是否必选	参数类型	描述
fields	否	Array	显示字段列表 数组长度：1 - 5

请求参数

表 5-3 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。获取Token，请参考《统一身份认证服务API参考》的“获取用户Token”章节。请求响应成功后在响应消息头中包含的“X-Subject-Token”的值即为Token值。 最小长度：0 最大长度：10240

响应参数

状态码：200

表 5-4 响应 Body 参数

参数	参数类型	描述
request_id	String	操作请求ID
direct_connect	DirectConnect object	物理专线对象

表 5-5 DirectConnect

参数	参数类型	描述
id	String	物理专线标识符ID
tenant_id	String	实例所属项目ID。
name	String	物理专线名字 最小长度：0 最大长度：64

参数	参数类型	描述
description	String	物理专线描述信息 最小长度：0 最大长度：128
port_type	String	物理专线接入接口的类型，支持1G 10G 40G 100G 枚举值： <ul style="list-style-type: none">• 1G• 10G• 40G• 100G
bandwidth	Integer	物理专线接入带宽，单位Mbps。 最小值：2 最大值：100000
location	String	专线的接入位置信息 最小长度：0 最大长度：255
peer_location	String	物理专线对端所在的物理位置，省/市/街道或IDC名字。 最小长度：0 最大长度：255
device_id	String	物理专线连接的设备的标识ID 最小长度：0 最大长度：36
type	String	物理专线的类型，类型包括标准(standard)，运营专线(hosting)，托管专线(hosted)。 缺省值：standard 枚举值： <ul style="list-style-type: none">• standard• hosting• hosted• onestop_standard• onestop_hosted
hosting_id	String	hosted物理专线对应的hosting物理专线的ID

参数	参数类型	描述
charge_mode	String	计费模式: prepayment/bandwidth/traffic 枚举值: <ul style="list-style-type: none">● prepayment● bandwidth● traffic
provider	String	物理专线连接的线路运营商 如: 中国电信 中国联通 中国移动 中国其他 境外其他专线归属的运营商
admin_state_up	Boolean	管理状态: true或false 缺省值: true
vlan	Integer	为托管hosted物理专线分配的vlan。 最小值: 0 最大值: 3999
status	String	资源状态, 合法值是: ACTIVE: 专线已经开通完成且线路处于正常状态 DOWN: 专线对应的端口处于down的状态, 可能存在线路故障等异常。 BUILD: 申请专线正在施工建设中 ERROR: 专线配置异常, 请联系客服解决相关问题。 PENDING_DELETE: 正在删除 DELETED: 已删除 APPLY: 申请开通 DENY: 客户需求无法满足, 拒绝工勘。 PENDING_PAY: 待支付 PAID: 已支付 PENDING_SURVEY: 待工勘 枚举值: <ul style="list-style-type: none">● BUILD● PAID● APPLY● PENDING_SURVEY● ACTIVE● DOWN● ERROR● PENDING_DELETE● DELETED● DENY● PENDING_PAY
apply_time	String	物理专线的申请时间。采用UTC时间格式, 格式为: yyyy-MM-ddTHH:mm:ss.SSSZ
create_time	String	物理专线的创建时间。采用UTC时间格式, 格式为: yyyy-MM-ddTHH:mm:ss.SSSZ

参数	参数类型	描述
provider_status	String	物理专线的运营商操作状态，合法值是： ACTIVE， DOWN 枚举值： <ul style="list-style-type: none">• ACTIVE• DOWN
peer_port_type	String	连接对端的端口类型
peer_provider	String	专线连接对接的运营商
order_id	String	物理专线对应订单号，用于支持包周期计费，识别用户订单
product_id	String	物理专线订单对应产品标识，用于订制包周期套餐等计费策略
spec_code	String	物理专线订单对应产品规格，用于订制包周期套餐等计费策略
period_type	Integer	物理专线对应订单号对应包周期的类型
period_num	Integer	物理专线对应的包周期时间
vgw_type	String	专线要求的网关类型 缺省值： default 枚举值： <ul style="list-style-type: none">• default
lag_id	String	物理专线归属的链路聚合组（lag）的ID
signed_agreement_status	String	专线协议的签署状态 枚举值： <ul style="list-style-type: none">• signed
signed_agreement_time	String	专线协议的签署时间
enterprise_project_id	String	实例所属企业项目ID 最小长度： 36 最大长度： 36
tags	Array of Tag objects	标签信息 数组长度： 0 - 10
locales	LocalesBody object	物理专线的区域信息
support_feature	Array of strings	用户专线可支持的特性列表
ies_id	String	归属的IES站点的ID

参数	参数类型	描述
reason	String	如果专线资源的状态是Error的情况下，该参数会显示相关错误信息。
email	String	客户邮箱信息
onestop_product_id	String	该参数用于销售线路场景，标识一站式专线产品ID
building_line_product_id	String	该参数用于销售线路场景，标识机房内部线路资源产品ID
last_onestop_product_id	String	该参数用于销售线路场景，标识变更前的一站式专线产品ID，用于在做线路带宽变更时保存上一次的记录。
last_building_line_product_id	String	该参数用于销售线路场景，标识变更前机房内部线路资源产品ID，用于在做线路带宽变更时保存上一次的记录。
modified_bandwidth	Integer	线路带宽变更后的带宽值
change_mode	Integer	标识续费变更的一种状态
onestopdc_status	String	一站式专线状态
public_border_group	String	归属的可用区对应的边界组(public border group)，标识是否homezone局点。
auto_renew	Integer	用于标识包周期产品是否自动续订
ratio_95peak	Integer	95计费保底带宽率 最小值： 0 最大值： 100

表 5-6 Tag

参数	参数类型	描述
key	String	标签键，最大长度36个unicode字符，格式为大小写字母，数字，中划线“-”，下划线“_”，中文。 最小长度： 0 最大长度： 36
value	String	标签值，最大长度43个unicode字符，格式为大小写字母，数字，中划线“-”，下划线“_”，点“.”，中文。 最小长度： 0 最大长度： 43

表 5-7 LocalesBody

参数	参数类型	描述
en_us	String	区域英文名称。 最小长度：0 最大长度：255
zh_cn	String	区域中文名称。 最小长度：0 最大长度：255

请求示例

查询物理专线详情

```
GET https://{dc_endpoint}/v3/6f9e9263116a4b68818cf1edce16bc4f/dcaas/direct-connects/6ecd9cf3-ca64-46c7-863f-f2eb1b9e838a
```

响应示例

状态码：200

OK

- 成功查询物理连接详细信息

```
{
  "direct_connect": {
    "bandwidth": 100,
    "create_time": "2018-10-19T09:53:26.000Z",
    "port_type": "10G",
    "id": "6ecd9cf3-ca64-46c7-863f-f2eb1b9e838a",
    "apply_time": "2018-10-19T09:53:26.000Z",
    "peer_location": "",
    "peer_port_type": null,
    "peer_provider": null,
    "location": "ExampleLocation",
    "provider": "ExampleProvider",
    "type": "standard",
    "status": "BUILD",
    "description": "",
    "provider_status": "ACTIVE",
    "order_id": "",
    "vlan": null,
    "device_id": "172.16.40.2",
    "name": "direct connect1",
    "admin_state_up": true,
    "tenant_id": "6f9e9263116a4b68818cf1edce16bc4f",
    "hosting_id": null,
    "product_id": "",
    "vgw_type": "default",
    "spec_code": "100ge",
    "charge_mode": null,
    "support_feature": [ ],
    "ies_id": null,
    "reason": null,
    "email": "cloud@example.com",
    "onestop_product_id": null,
  }
}
```

```
"building_line_product_id" : null,  
"last_building_line_product_id" : null,  
"last_onestop_product_id" : null,  
"modified_bandwidth" : null,  
"change_mode" : null,  
"onestopdc_status" : null,  
"public_border_group" : "center",  
"auto_renew" : 0,  
"ratio_95peak" : null  
}  
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;  
  
import com.huaweicloud.sdk.core.auth.ICredential;  
import com.huaweicloud.sdk.core.auth.BasicCredentials;  
import com.huaweicloud.sdk.core.exception.ConnectionException;  
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;  
import com.huaweicloud.sdk.core.exception.ServiceResponseException;  
import com.huaweicloud.sdk.dc.v3.region.DcRegion;  
import com.huaweicloud.sdk.dc.v3.*;  
import com.huaweicloud.sdk.dc.v3.model.*;  
  
import java.util.List;  
import java.util.ArrayList;  
  
public class ShowDirectConnectSolution {  
    public static void main(String[] args) {  
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great  
        // security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or  
        // environment variables and decrypted during use to ensure security.  
        // In this example, AK and SK are stored in environment variables for authentication. Before running  
        // this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment  
        String ak = System.getenv("CLOUD_SDK_AK");  
        String sk = System.getenv("CLOUD_SDK_SK");  
  
        ICredential auth = new BasicCredentials()  
            .withAk(ak)  
            .withSk(sk);  
  
        DcClient client = DcClient.newBuilder()  
            .withCredential(auth)  
            .withRegion(DcRegion.valueOf("<YOUR REGION>"))  
            .build();  
        ShowDirectConnectRequest request = new ShowDirectConnectRequest();  
        request.withFields();  
        try {  
            ShowDirectConnectResponse response = client.showDirectConnect(request);  
            System.out.println(response.toString());  
        } catch (ConnectionException e) {  
            e.printStackTrace();  
        } catch (RequestTimeoutException e) {  
            e.printStackTrace();  
        } catch (ServiceResponseException e) {  
            e.printStackTrace();  
            System.out.println(e.getHttpStatusCode());  
            System.out.println(e.getRequestId());  
            System.out.println(e.getErrorCode());  
            System.out.println(e.getErrorMsg());  
        }  
    }  
}
```

```
}  
}
```

Python

```
# coding: utf-8  
  
from huaweicloudsdkcore.auth.credentials import BasicCredentials  
from huaweicloudsdkdc.v3.region.dc_region import DcRegion  
from huaweicloudsdkcore.exceptions import exceptions  
from huaweicloudsdkdc.v3 import *  
  
if __name__ == "__main__":  
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security  
    # risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment  
    # variables and decrypted during use to ensure security.  
    # In this example, AK and SK are stored in environment variables for authentication. Before running this  
    # example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment  
    ak = os.getenv("CLOUD_SDK_AK")  
    sk = os.getenv("CLOUD_SDK_SK")  
  
    credentials = BasicCredentials(ak, sk) \  
  
    client = DcClient.new_builder() \  
        .with_credentials(credentials) \  
        .with_region(DcRegion.value_of("<YOUR REGION>")) \  
        .build()  
  
    try:  
        request = ShowDirectConnectRequest()  
        request.fields = \  
        response = client.show_direct_connect(request)  
        print(response)  
    except exceptions.ClientRequestException as e:  
        print(e.status_code)  
        print(e.request_id)  
        print(e.error_code)  
        print(e.error_msg)
```

Go

```
package main  
  
import (  
    "fmt"  
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"  
    dc "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3"  
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/model"  
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/region"  
)  
  
func main() {  
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security  
    // risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment  
    // variables and decrypted during use to ensure security.  
    // In this example, AK and SK are stored in environment variables for authentication. Before running this  
    // example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment  
    ak := os.Getenv("CLOUD_SDK_AK")  
    sk := os.Getenv("CLOUD_SDK_SK")  
  
    auth := basic.NewCredentialsBuilder().  
        WithAk(ak).  
        WithSk(sk).  
        Build()  
  
    client := dc.NewDcClient(  
        dc.DcClientBuilder().  
            WithRegion(region.ValueOf("<YOUR REGION>")).  
            WithCredential(auth).
```



```
Build()

request := &model.ShowDirectConnectRequest{}
response, err := client.ShowDirectConnect(request)
if err == nil {
    fmt.Printf("%+v\n", response)
} else {
    fmt.Println(err)
}
}
```

更多

更多编程语言的SDK代码示例，请参见[API Explorer](#)的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK

错误码

请参见[错误码](#)。

5.1.2 更新物理连接信息

功能介绍

更新物理连接信息，包括名字,描述等信息

调用方法

请参见[如何调用API](#)。

URI

PUT /v3/{project_id}/dcaas/direct-connects/{direct_connect_id}

表 5-8 路径参数

参数	是否必选	参数类型	描述
direct_connect_id	是	String	物理专线连接ID。 最小长度：36 最大长度：36
project_id	是	String	租户项目ID

请求参数

表 5-9 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。获取Token，请参考《统一身份认证服务API参考》的“获取用户Token”章节。请求响应成功后在响应消息头中包含的“X-Subject-Token”的值即为Token值。 最小长度：0 最大长度：10240

表 5-10 请求 Body 参数

参数	是否必选	参数类型	描述
direct_connect	否	UpdateDirectConnect object	物理专线更新参数

表 5-11 UpdateDirectConnect

参数	是否必选	参数类型	描述
name	否	String	物理专线的名字 最小长度：0 最大长度：64
description	否	String	物理专线的描述信息 最小长度：0 最大长度：128
bandwidth	否	Integer	指定托管专线接入带宽,单位Mbps。 最小值：2 最大值：100000
peer_location	否	String	物理专线对端所在的物理位置,省/市/街道或IDC名字 最小长度：0 最大长度：255

参数	是否必选	参数类型	描述
status	否	String	更新资源状态，合法值是： PENDING_PAY, APPLY 枚举值： <ul style="list-style-type: none">● PENDING_PAY● APPLY
provider_status	否	String	更新运营商状态，合法值是： ACTIVE,DOWN 枚举值： <ul style="list-style-type: none">● ACTIVE● DOWN

响应参数

状态码： 200

表 5-12 响应 Body 参数

参数	参数类型	描述
request_id	String	操作请求ID
direct_connect	DirectConnect object	物理专线对象

表 5-13 DirectConnect

参数	参数类型	描述
id	String	物理专线标识符ID
tenant_id	String	实例所属项目ID。
name	String	物理专线名字 最小长度： 0 最大长度： 64
description	String	物理专线描述信息 最小长度： 0 最大长度： 128

参数	参数类型	描述
port_type	String	物理专线接入接口的类型，支持1G 10G 40G 100G 枚举值： <ul style="list-style-type: none">• 1G• 10G• 40G• 100G
bandwidth	Integer	物理专线接入带宽，单位Mbps。 最小值：2 最大值：100000
location	String	专线的接入位置信息 最小长度：0 最大长度：255
peer_location	String	物理专线对端所在的物理位置，省/市/街道或IDC名字。 最小长度：0 最大长度：255
device_id	String	物理专线连接的设备的标识ID 最小长度：0 最大长度：36
type	String	物理专线的类型，类型包括标准(standard)，运营专线(hosting)，托管专线(hosted)。 缺省值： standard 枚举值： <ul style="list-style-type: none">• standard• hosting• hosted• onestop_standard• onestop_hosted
hosting_id	String	hosted物理专线对应的hosting物理专线的ID
charge_mode	String	计费模式：prepayment/bandwidth/traffic 枚举值： <ul style="list-style-type: none">• prepayment• bandwidth• traffic

参数	参数类型	描述
provider	String	物理专线连接的线路运营商 如：中国电信 中国联通 中国移动 中国其他 境外其他专线归属的运营商
admin_state_up	Boolean	管理状态：true或false 缺省值： true
vlan	Integer	为托管hosted物理专线分配的vlan。 最小值： 0 最大值： 3999
status	String	资源状态，合法值是：ACTIVE：专线已经开通完成且线路处于正常状态 DOWN：专线对应的端口处于down的状态，可能存在线路故障等异常。 BUILD：申请专线正在施工建设中 ERROR：专线配置异常，请联系客服解决相关问题。 PENDING_DELETE：正在删除 DELETED：已删除 APPLY：申请开通 DENY：客户需求无法满足，拒绝工勘。 PENDING_PAY：待支付 PAID：已支付 PENDING_SURVEY：待工勘 枚举值： <ul style="list-style-type: none">● BUILD● PAID● APPLY● PENDING_SURVEY● ACTIVE● DOWN● ERROR● PENDING_DELETE● DELETED● DENY● PENDING_PAY
apply_time	String	物理专线的申请时间。采用UTC时间格式，格式为：yyyy-MM-ddTHH:mm:ss.SSSZ
create_time	String	物理专线的创建时间。采用UTC时间格式，格式为：yyyy-MM-ddTHH:mm:ss.SSSZ
provider_statuses	String	物理专线的运营商操作状态，合法值是：ACTIVE， DOWN 枚举值： <ul style="list-style-type: none">● ACTIVE● DOWN

参数	参数类型	描述
peer_port_type	String	连接对端的端口类型
peer_provider	String	专线连接对接的运营商
order_id	String	物理专线对应订单号，用于支持包周期计费，识别用户订单
product_id	String	物理专线订单对应产品标识，用于订制包周期套餐等计费策略
spec_code	String	物理专线订单对应产品规格，用于订制包周期套餐等计费策略
period_type	Integer	物理专线对应订单号对应包周期的类型
period_num	Integer	物理专线对应的包周期时间
vgw_type	String	专线要求的网关类型 缺省值： default 枚举值： <ul style="list-style-type: none">• default
lag_id	String	物理专线归属的链路聚合组（lag）的ID
signed_agreement_status	String	专线协议的签署状态 枚举值： <ul style="list-style-type: none">• signed
signed_agreement_time	String	专线协议的签署时间
enterprise_project_id	String	实例所属企业项目ID 最小长度： 36 最大长度： 36
tags	Array of Tag objects	标签信息 数组长度： 0 - 10
locales	LocalesBody object	物理专线的区域信息
support_feature	Array of strings	用户专线可支持的特性列表
ies_id	String	归属的IES站点的ID
reason	String	如果专线资源的状态是Error的情况下，该参数会显示相关错误信息。
email	String	客户邮箱信息

参数	参数类型	描述
onestop_product_id	String	该参数用于销售线路场景，标识一站式专线产品ID
building_line_product_id	String	该参数用于销售线路场景，标识机房内部线路资源产品ID
last_onestop_product_id	String	该参数用于销售线路场景，标识变更前的一站式专线产品ID，用于在做线路带宽变更时保存上一次的记录。
last_building_line_product_id	String	该参数用于销售线路场景，标识变更前机房内部线路资源产品ID，用于在做线路带宽变更时保存上一次的记录。
modified_bandwidth	Integer	线路带宽变更后的带宽值
change_mode	Integer	标识续费变更的一种状态
onestopdc_status	String	一站式专线状态
public_border_group	String	归属的可用区对应的边界组(public border group)，标识是否homezone局点。
auto_renew	Integer	用于标识包周期产品是否自动续订
ratio_95peak	Integer	95计费保底带宽率 最小值： 0 最大值： 100

表 5-14 Tag

参数	参数类型	描述
key	String	标签键，最大长度36个unicode字符，格式为大小写字母，数字，中划线“-”，下划线“_”，中文。 最小长度： 0 最大长度： 36
value	String	标签值，最大长度43个unicode字符，格式为大小写字母，数字，中划线“-”，下划线“_”，点“.”，中文。 最小长度： 0 最大长度： 43

表 5-15 LocalesBody

参数	参数类型	描述
en_us	String	区域英文名称。 最小长度：0 最大长度：255
zh_cn	String	区域中文名称。 最小长度：0 最大长度：255

请求示例

更新物理连接的名称和描述信息

PUT https://{dc_endpoint}/v3/6fbc9263116a4b68818cf1edce16bc4f/dcaas/direct-connects/6ecd9cf3-ca64-46c7-863f-f2eb1b9e838a

```
{
  "direct_connect": {
    "description": "",
    "name": "direct connect1"
  }
}
```

响应示例

状态码：200

OK

- 成功更新物理连接信息

```
{
  "request_id": "9a4f4dfc4fb2fc101e65bba07d908535",
  "direct_connect": {
    "bandwidth": 100,
    "create_time": "2018-10-19T09:53:26.000Z",
    "port_type": "10G",
    "id": "6ecd9cf3-ca64-46c7-863f-f2eb1b9e838a",
    "apply_time": "2018-10-19T09:53:26.000Z",
    "peer_location": "",
    "peer_port_type": null,
    "peer_provider": null,
    "location": "ExampleLocation",
    "provider": "ExampleProvider",
    "type": "standard",
    "status": "BUILD",
    "description": "",
    "provider_status": "ACTIVE",
    "order_id": "",
    "vlan": null,
    "device_id": "172.16.40.2",
    "name": "direct connect1",
    "admin_state_up": true,
    "tenant_id": "6fbc9263116a4b68818cf1edce16bc4f",
    "hosting_id": null,
    "product_id": "",
    "vgw_type": "default",
    "spec_code": "100ge",
    "charge_mode": null,
  }
}
```



```
"support_feature": [ ],
"ies_id": null,
"reason": null,
"email": "cloud@example.com",
"onestop_product_id": null,
"building_line_product_id": null,
"last_building_line_product_id": null,
"last_onestop_product_id": null,
"modified_bandwidth": null,
"change_mode": null,
"onestopdc_status": null,
"public_border_group": "center",
"auto_renew": 0,
"ratio_95peak": null
}
}
```

SDK 代码示例

SDK代码示例如下。

Java

更新物理连接的名称和描述信息

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.dc.v3.region.DcRegion;
import com.huaweicloud.sdk.dc.v3.*;
import com.huaweicloud.sdk.dc.v3.model.*;

public class UpdateDirectConnectSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
        // security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
        // environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running
        // this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
            .withAk(ak)
            .withSk(sk);

        DcClient client = DcClient.newBuilder()
            .withCredential(auth)
            .withRegion(DcRegion.valueOf("<YOUR REGION>"))
            .build();
        UpdateDirectConnectRequest request = new UpdateDirectConnectRequest();
        UpdateDirectConnectRequestBody body = new UpdateDirectConnectRequestBody();
        UpdateDirectConnect directConnectbody = new UpdateDirectConnect();
        directConnectbody.setName("direct connect1")
            .withDescription("");
        body.withDirectConnect(directConnectbody);
        request.withBody(body);
        try {
            UpdateDirectConnectResponse response = client.updateDirectConnect(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
```

```
        e.printStackTrace();
    } catch (RequestTimeoutException e) {
        e.printStackTrace();
    } catch (ServiceResponseException e) {
        e.printStackTrace();
        System.out.println(e.getHttpStatusCode());
        System.out.println(e.getRequestId());
        System.out.println(e.getErrorCode());
        System.out.println(e.getErrorMsg());
    }
}
}
```

Python

更新物理连接的名称和描述信息

```
# coding: utf-8

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdc.v3.region.dc_region import DcRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdc.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    # risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    # variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this
    # example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.getenv("CLOUD_SDK_AK")
    sk = os.getenv("CLOUD_SDK_SK")

    credentials = BasicCredentials(ak, sk) \

    client = DcClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DcRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = UpdateDirectConnectRequest()
        directConnectbody = UpdateDirectConnect(
            name="direct connect1",
            description=""
        )
        request.body = UpdateDirectConnectRequestBody(
            direct_connect=directConnectbody
        )
        response = client.update_direct_connect(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

更新物理连接的名称和描述信息

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dc "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/model"
```

```
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    // risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    // variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this
    // example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := dc.NewDcClient(
        dc.DcClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.UpdateDirectConnectRequest{
        nameDirectConnect:= "direct connect1"
        descriptionDirectConnect:= ""
        directConnectbody := &model.UpdateDirectConnect{
            Name: &nameDirectConnect,
            Description: &descriptionDirectConnect,
        }
        request.Body = &model.UpdateDirectConnectRequestBody{
            DirectConnect: directConnectbody,
        }
    }
    response, err := client.UpdateDirectConnect(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见[API Explorer](#)的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK

错误码

请参见[错误码](#)。

5.1.3 删除物理连接

功能介绍

删除物理连接，本接口只适用于按需计费物理专线，对于包周期购买的专线通过订单退订的方式删除物理连接

调用方法

请参见[如何调用API](#)。

URI

DELETE /v3/{project_id}/dcaas/direct-connects/{direct_connect_id}

表 5-16 路径参数

参数	是否必选	参数类型	描述
direct_connect_id	是	String	物理专线连接ID。 最小长度：36 最大长度：36
project_id	是	String	租户项目ID

请求参数

表 5-17 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。获取Token，请参考《统一身份认证服务API参考》的“获取用户Token”章节。请求响应成功后在响应消息头中包含的“X-Subject-Token”的值即为Token值。 最小长度：0 最大长度：10240

响应参数

无

请求示例

删除物理连接

```
DELETE https://{dc_endpoint}/v3/6f9e263116a4b68818cf1edce16bc4f/dcaas/direct-connects/6ecd9cf3-ca64-46c7-863f-f2eb1b9e838a
```

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.dc.v3.region.DcRegion;
import com.huaweicloud.sdk.dc.v3.*;
import com.huaweicloud.sdk.dc.v3.model.*;

public class DeleteDirectConnectSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
        // security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
        // environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running
        // this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
            .withAk(ak)
            .withSk(sk);

        DcClient client = DcClient.newBuilder()
            .withCredential(auth)
            .withRegion(DcRegion.valueOf("<YOUR REGION>"))
            .build();
        DeleteDirectConnectRequest request = new DeleteDirectConnectRequest();
        try {
            DeleteDirectConnectResponse response = client.deleteDirectConnect(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8
```

```
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdc.v3.region.dc_region import DcRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdc.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    # risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    # variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this
    # example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.getenv("CLOUD_SDK_AK")
    sk = os.getenv("CLOUD_SDK_SK")

    credentials = BasicCredentials(ak, sk) \

    client = DcClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DcRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = DeleteDirectConnectRequest()
        response = client.delete_direct_connect(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dc "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    // risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    // variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this
    // example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := dc.NewDcClient(
        dc.DcClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.DeleteDirectConnectRequest{}
    response, err := client.DeleteDirectConnect(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

```
}  
}
```

更多

更多编程语言的SDK代码示例，请参见[API Explorer](#)的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
204	No Content

错误码

请参见[错误码](#)。

5.1.4 查询物理连接列表

功能介绍

查询租户创建的所有的direct connect对象。

调用方法

请参见[如何调用API](#)。

URI

GET /v3/{project_id}/dcaas/direct-connects

表 5-18 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	租户项目ID

表 5-19 Query 参数

参数	是否必选	参数类型	描述
limit	否	Integer	每页返回的个数。取值范围： 1~2000。 最小值：1 最大值：2000 缺省值：2000

参数	是否必选	参数类型	描述
marker	否	String	上一页最后一条资源记录的ID，为空时为查询第一页。使用说明：必须与limit一起使用。 最小长度：0 最大长度：36
fields	否	Array	显示字段列表 数组长度：1 - 5
sort_key	否	String	排序字段。 缺省值：id 最小长度：0 最大长度：36
sort_dir	否	Array	返回结果按照升序(asc)或降序(desc)排列，默认为asc
hosting_id	否	Array	根据运营专线ID过滤托管专线列表 数组长度：0 - 5
enterprise_project_id	否	Array	根据企业项目ID过滤资源实例 数组长度：1 - 10
id	否	Array	根据资源ID过滤实例 数组长度：1 - 5
name	否	Array	根据名字过滤查询，可查询多个名字。 数组长度：1 - 5

请求参数

表 5-20 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。获取Token，请参考《统一身份认证服务API参考》的“获取用户Token”章节。请求响应成功后在响应消息头中包含的“X-Subject-Token”的值即为Token值。 最小长度：0 最大长度：10240

响应参数

状态码： 200

表 5-21 响应 Body 参数

参数	参数类型	描述
request_id	String	操作请求ID
direct_connects	Array of DirectConnect objects	物理专线对象列表
page_info	PageInfo object	分页查询页的信息

表 5-22 DirectConnect

参数	参数类型	描述
id	String	物理专线标识符ID
tenant_id	String	实例所属项目ID。
name	String	物理专线名字 最小长度： 0 最大长度： 64
description	String	物理专线描述信息 最小长度： 0 最大长度： 128
port_type	String	物理专线接入接口的类型，支持1G 10G 40G 100G 枚举值： <ul style="list-style-type: none">• 1G• 10G• 40G• 100G
bandwidth	Integer	物理专线接入带宽，单位Mbps。 最小值： 2 最大值： 100000
location	String	专线的接入位置信息 最小长度： 0 最大长度： 255

参数	参数类型	描述
peer_location	String	物理专线对端所在的物理位置，省/市/街道或IDC名字。 最小长度：0 最大长度：255
device_id	String	物理专线连接的设备的标识ID 最小长度：0 最大长度：36
type	String	物理专线的类型，类型包括标准(standard)，运营专线(hosting)，托管专线(hosted)。 缺省值： standard 枚举值： <ul style="list-style-type: none">• standard• hosting• hosted• onestop_standard• onestop_hosted
hosting_id	String	hosted物理专线对应的hosting物理专线的ID
charge_mode	String	计费模式：prepayment/bandwidth/traffic 枚举值： <ul style="list-style-type: none">• prepayment• bandwidth• traffic
provider	String	物理专线连接的线路运营商 如：中国电信 中国联通 中国移动 中国其他 境外其他专线归属的运营商
admin_state_up	Boolean	管理状态：true或false 缺省值： true
vlan	Integer	为托管hosted物理专线分配的vlan。 最小值：0 最大值： 3999

参数	参数类型	描述
status	String	资源状态，合法值是：ACTIVE：专线已经开通完成且线路处于正常状态 DOWN：专线对应的端口处于down的状态，可能存在线路故障等异常。BUILD：申请专线正在施工建设中 ERROR：专线配置异常，请联系客服解决相关问题。PENDING_DELETE：正在删除 DELETED：已删除 APPLY：申请开通 DENY：客户需求无法满足，拒绝工勘。PENDING_PAY：待支付 PAID：已支付 PENDING_SURVEY：待工勘 枚举值： <ul style="list-style-type: none">● BUILD● PAID● APPLY● PENDING_SURVEY● ACTIVE● DOWN● ERROR● PENDING_DELETE● DELETED● DENY● PENDING_PAY
apply_time	String	物理专线的申请时间。采用UTC时间格式，格式为：yyyy-MM-ddTHH:mm:ss.SSSZ
create_time	String	物理专线的创建时间。采用UTC时间格式，格式为：yyyy-MM-ddTHH:mm:ss.SSSZ
provider_status	String	物理专线的运营商操作状态，合法值是：ACTIVE， DOWN 枚举值： <ul style="list-style-type: none">● ACTIVE● DOWN
peer_port_type	String	连接对端的端口类型
peer_provider	String	专线连接对接的运营商
order_id	String	物理专线对应订单号，用于支持包周期计费，识别用户订单
product_id	String	物理专线订单对应产品标识，用于订制包周期套餐等计费策略
spec_code	String	物理专线订单对应产品规格，用于订制包周期套餐等计费策略

参数	参数类型	描述
period_type	Integer	物理专线对应订单号对应包周期的类型
period_num	Integer	物理专线对应的包周期时间
vgw_type	String	专线要求的网关类型 缺省值： default 枚举值： <ul style="list-style-type: none">• default
lag_id	String	物理专线归属的链路聚合组（lag）的ID
signed_agreement_status	String	专线协议的签署状态 枚举值： <ul style="list-style-type: none">• signed
signed_agreement_time	String	专线协议的签署时间
enterprise_project_id	String	实例所属企业项目ID 最小长度： 36 最大长度： 36
tags	Array of Tag objects	标签信息 数组长度： 0 - 10
locales	LocalesBody object	物理专线的区域信息
support_feature	Array of strings	用户专线可支持的特性列表
ies_id	String	归属的IES站点的ID
reason	String	如果专线资源的状态是Error的情况下，该参数会显示相关错误信息。
email	String	客户邮箱信息
onestop_product_id	String	该参数用于销售线路场景，标识一站式专线产品ID
building_line_product_id	String	该参数用于销售线路场景，标识机房内部线路资源产品ID
last_onestop_product_id	String	该参数用于销售线路场景，标识变更前的一站式专线产品ID，用于在做线路带宽变更时保存上一次的记录。
last_building_line_product_id	String	该参数用于销售线路场景，标识变更前机房内部线路资源产品ID，用于在做线路带宽变更时保存上一次的记录。

参数	参数类型	描述
modified_bandwidth	Integer	线路带宽变更后的带宽值
change_mode	Integer	标识续费变更的一种状态
onestopdc_status	String	一站式专线状态
public_border_group	String	归属的可用区对应的边界组(public border group)，标识是否homezone局点。
auto_renew	Integer	用于标识包周期产品是否自动续订
ratio_95peak	Integer	95计费保底带宽率 最小值： 0 最大值： 100

表 5-23 Tag

参数	参数类型	描述
key	String	标签键，最大长度36个unicode字符，格式为大小写字母，数字，中划线“-”，下划线“_”，中文。 最小长度： 0 最大长度： 36
value	String	标签值，最大长度43个unicode字符，格式为大小写字母，数字，中划线“-”，下划线“_”，点“.”，中文。 最小长度： 0 最大长度： 43

表 5-24 LocalesBody

参数	参数类型	描述
en_us	String	区域英文名称。 最小长度： 0 最大长度： 255
zh_cn	String	区域中文名称。 最小长度： 0 最大长度： 255

表 5-25 PageInfo

参数	参数类型	描述
previous_marker	String	上一页的marker, 值为资源的uuid 最小长度: 0 最大长度: 36
current_count	Integer	当前列表中资源数量 最小值: 0 最大值: 2000
next_marker	String	下一页的marker, 值为资源的uuid, 为空时表示最后一页 最小长度: 0 最大长度: 36

请求示例

查询物理连接列表

```
GET https://{dc_endpoint}/v3/6fbe9263116a4b68818cf1edce16bc4f/dcaas/direct-connects
```

响应示例

状态码: 200

OK

- 成功查询物理连接列表

```
{
  "request_id": "9a4f4dfc4fb2fc101e65bba07d908535",
  "direct_connects": [ {
    "bandwidth": 100,
    "create_time": "2018-10-19T09:53:26.000Z",
    "port_type": "10G",
    "id": "6ecd9cf3-ca64-46c7-863f-f2eb1b9e838a",
    "apply_time": "2018-10-19T09:53:26.000Z",
    "peer_location": "",
    "peer_port_type": null,
    "peer_provider": null,
    "location": "ExampleLocation",
    "provider": "ExampleProvider",
    "type": "standard",
    "status": "BUILD",
    "description": "",
    "provider_status": "ACTIVE",
    "order_id": "",
    "vlan": null,
    "device_id": "172.16.40.2",
    "name": "direct connect1",
    "admin_state_up": true,
    "tenant_id": "6fbe9263116a4b68818cf1edce16bc4f",
    "hosting_id": null,
    "product_id": "",
    "vgw_type": "default",
    "spec_code": "100ge",
    "charge_mode": null,
    "support_feature": [ ],
```

```
"ies_id" : null,  
"reason" : null,  
"email" : "cloud@example.com",  
"onestop_product_id" : null,  
"building_line_product_id" : null,  
"last_building_line_product_id" : null,  
"last_onestop_product_id" : null,  
"modified_bandwidth" : null,  
"change_mode" : null,  
"onestopdc_status" : null,  
"public_border_group" : "center",  
"auto_renew" : 0,  
"ratio_95peak" : null  
} ]  
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;  
  
import com.huaweicloud.sdk.core.auth.ICredential;  
import com.huaweicloud.sdk.core.auth.BasicCredentials;  
import com.huaweicloud.sdk.core.exception.ConnectionException;  
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;  
import com.huaweicloud.sdk.core.exception.ServiceResponseException;  
import com.huaweicloud.sdk.dc.v3.region.DcRegion;  
import com.huaweicloud.sdk.dc.v3.*;  
import com.huaweicloud.sdk.dc.v3.model.*;  
  
import java.util.List;  
import java.util.ArrayList;  
  
public class ListDirectConnectsSolution {  
  
    public static void main(String[] args) {  
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great  
        // security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or  
        // environment variables and decrypted during use to ensure security.  
        // In this example, AK and SK are stored in environment variables for authentication. Before running  
        // this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment  
        String ak = System.getenv("CLOUD_SDK_AK");  
        String sk = System.getenv("CLOUD_SDK_SK");  
  
        ICredential auth = new BasicCredentials()  
            .withAk(ak)  
            .withSk(sk);  
  
        DcClient client = DcClient.newBuilder()  
            .withCredential(auth)  
            .withRegion(DcRegion.valueOf("<YOUR REGION>"))  
            .build();  
        ListDirectConnectsRequest request = new ListDirectConnectsRequest();  
        request.withLimit(<limit>);  
        request.withMarker("<marker>");  
        request.withFields();  
        request.withSortKey("<sort_key>");  
        request.withSortDir();  
        request.withHostingId();  
        request.withEnterpriseProjectId();  
        request.withId();  
        request.withName();  
        try {  
            ListDirectConnectsResponse response = client.listDirectConnects(request);  
            System.out.println(response.toString());  
        }  
    }  
}
```

```
    } catch (ConnectionException e) {
        e.printStackTrace();
    } catch (RequestTimeoutException e) {
        e.printStackTrace();
    } catch (ServiceResponseException e) {
        e.printStackTrace();
        System.out.println(e.getHttpStatusCode());
        System.out.println(e.getRequestId());
        System.out.println(e.getErrorCode());
        System.out.println(e.getErrorMsg());
    }
}
}
```

Python

```
# coding: utf-8

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdc.v3.region.dc_region import DcRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdc.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    # risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    # variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this
    # example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.getenv("CLOUD_SDK_AK")
    sk = os.getenv("CLOUD_SDK_SK")

    credentials = BasicCredentials(ak, sk) \

    client = DcClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DcRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListDirectConnectsRequest()
        request.limit = <limit>
        request.marker = "<marker>"
        request.fields =
        request.sort_key = "<sort_key>"
        request.sort_dir =
        request.hosting_id =
        request.enterprise_project_id =
        request.id =
        request.name =
        response = client.list_direct_connects(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dc "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/region"
)
```



```
func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    // risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    // variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this
    // example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := dc.NewDcClient(
        dc.DcClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ListDirectConnectsRequest{}
    limitRequest:= int32(<limit>)
    request.Limit = &limitRequest
    markerRequest:= "<marker>"
    request.Marker = &markerRequest
    sortKeyRequest:= "<sort_key>"
    request.SortKey = &sortKeyRequest
    response, err := client.ListDirectConnects(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见[API Explorer](#)的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK

错误码

请参见[错误码](#)。

5.1.5 查询租户的托管专线列表

功能介绍

查询合作伙伴创建的托管专线连接列表。

调用方法

请参见[如何调用API](#)。

URI

GET /v3/{project_id}/dcaas/hosted-connects

表 5-26 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	租户项目ID

表 5-27 Query 参数

参数	是否必选	参数类型	描述
limit	否	Integer	每页返回的个数。取值范围： 1~2000。 最小值：1 最大值：2000 缺省值：2000
marker	否	String	上一页最后一条资源记录的ID， 为空时为查询第一页。使用说明： 必须与limit一起使用。 最小长度：0 最大长度：36
fields	否	Array	显示字段列表 数组长度：1 - 5
sort_dir	否	Array	返回结果按照升序(asc)或降序 (desc)排列，默认为asc
sort_key	否	String	排序字段。 缺省值：id 最小长度：0 最大长度：36
hosting_id	否	Array	根据运营专线ID过滤托管专线列表 数组长度：0 - 5
id	否	Array	根据资源ID过滤实例 数组长度：1 - 5
name	否	Array	根据名字过滤查询，可查询多个 名字。 数组长度：1 - 5

请求参数

表 5-28 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。获取Token，请参考《统一身份认证服务API参考》的“获取用户Token”章节。请求响应成功后在响应消息头中包含的“X-Subject-Token”的值即为Token值。 最小长度：0 最大长度：10240

响应参数

状态码：200

表 5-29 响应 Body 参数

参数	参数类型	描述
request_id	String	本次操作的请求ID
hosted_connects	Array of HostedDirectConnect objects	托管专线对象信息
page_info	PageInfo object	分页查询页的信息

表 5-30 HostedDirectConnect

参数	参数类型	描述
id	String	托管专线ID 最小长度：36 最大长度：36
tenant_id	String	实例所属项目ID。
name	String	物理专线名字 最小长度：0 最大长度：64

参数	参数类型	描述
description	String	物理专线的描述信息 最小长度：0 最大长度：128
bandwidth	Integer	物理专线接入带宽，单位Mbps。 最小值：2 最大值：400000
location	String	专线的接入位置信息 最小长度：0 最大长度：255
peer_location	String	物理专线对端所在的物理位置，省/市/街道或IDC名字。 最小长度：0 最大长度：255
hosting_id	String	hosted物理专线对应的hosting物理专线的ID
provider	String	专线线路的提供商
admin_state_up	Boolean	管理状态：true或false 缺省值：true
vlan	Integer	hosted物理专线预分配的vlan。 最小值：0 最大值：3999
status	String	操作状态，合法值是：BUILD：已开通 ACTIVE：虚拟网关正常 DOWN：专线对应的端口处于down的状态，可能存在线路故障等异常。 ERROR：虚拟网关异常 PENDING_DELETE：删除中 PENDING_UPDATE：更新中 PENDING_CREATE：创建中 枚举值： <ul style="list-style-type: none">● BUILD● ACTIVE● DOWN● ERROR● PENDING_DELETE● PENDING_UPDATE● PENDING_CREATE
apply_time	String	物理专线申请时间。采用UTC时间格式，格式为：yyyy-MM-ddTHH:mm:ss.SSSZ

参数	参数类型	描述
create_time	String	物理专线创建时间。采用UTC时间格式，格式为：yyyy-MM-ddTHH:mm:ss.SSSZ
provider_statuses	String	物理专线的运营商操作状态，合法值是：ACTIVE，DOWN 枚举值： <ul style="list-style-type: none">● ACTIVE● DOWN
port_type	String	物理专线接入接口的类型，支持1G 10G 40G 100G 枚举值： <ul style="list-style-type: none">● 1G● 10G● 40G● 100G
type	String	物理专线的类型，类型包括标准(standard)，运营专线(hosting)，托管专线(hosted)。 缺省值： standard 枚举值： <ul style="list-style-type: none">● standard● hosting● hosted● onestop_standard● onestop_hosted

表 5-31 PageInfo

参数	参数类型	描述
previous_marker	String	上一页的marker，值为资源的uuid 最小长度： 0 最大长度： 36
current_count	Integer	当前列表中资源数量 最小值： 0 最大值： 2000
next_marker	String	下一页的marker，值为资源的uuid，为空时表示最后一页 最小长度： 0 最大长度： 36

请求示例

查询合作伙伴创建的托管专线列表

```
POST https://{dc_endpoint}/v3/6f9e9263116a4b68818cf1edce16bc4f/dcaas/hosted-connects
```

响应示例

状态码： 200

OK

- 成功查询托管专线连接列表

```
{
  "request_id": "a59a3776faa1d055f8124dc7b0977a90",
  "hosted_connects": [ {
    "id": "0278b472-ffa5-4eb3-8c0d-979d479f8ef6",
    "name": "client-dc-faf1",
    "description": "Hosted Connect",
    "tenant_id": "0605768a3300d5762f82c01180692873",
    "hosting_id": "2cfb53be-b05f-40d5-a2f8-3a59ac383836",
    "vlan": 441,
    "bandwidth": 10,
    "location": "ExampleLocation",
    "peer_location": "",
    "provider": "ExampleProvider",
    "type": "hosted",
    "port_type": "10G",
    "provider_status": "ACTIVE",
    "status": "ACTIVE",
    "apply_time": "2022-07-13T08:25:38.000Z",
    "admin_state_up": true,
    "create_time": "2022-07-13T08:25:38.000Z"
  } ]
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.dc.v3.region.DcRegion;
import com.huaweicloud.sdk.dc.v3.*;
import com.huaweicloud.sdk.dc.v3.model.*;

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

public class ListHostedDirectConnectsSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
        // security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
        // environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running
        // this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");
```

```
ICredential auth = new BasicCredentials()
    .withAk(ak)
    .withSk(sk);

DcClient client = DcClient.newBuilder()
    .withCredential(auth)
    .withRegion(DcRegion.valueOf("<YOUR REGION>"))
    .build();
ListHostedDirectConnectsRequest request = new ListHostedDirectConnectsRequest();
request.withLimit(<limit>);
request.withMarker("<marker>");
request.withFields();
request.withSortDir();
request.withSortKey("<sort_key>");
request.withHostingId();
request.withId();
request.withName();
try {
    ListHostedDirectConnectsResponse response = client.listHostedDirectConnects(request);
    System.out.println(response.toString());
} catch (ConnectionException e) {
    e.printStackTrace();
} catch (RequestTimeoutException e) {
    e.printStackTrace();
} catch (ServiceResponseException e) {
    e.printStackTrace();
    System.out.println(e.getHttpStatusCode());
    System.out.println(e.getRequestId());
    System.out.println(e.getErrorCode());
    System.out.println(e.getErrorMsg());
}
}
```

Python

```
# coding: utf-8

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdc.v3.region.dc_region import DcRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdc.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    # risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    # variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this
    # example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.getenv("CLOUD_SDK_AK")
    sk = os.getenv("CLOUD_SDK_SK")

    credentials = BasicCredentials(ak, sk) \

    client = DcClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DcRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListHostedDirectConnectsRequest()
        request.limit = <limit>
        request.marker = "<marker>"
        request.fields =
        request.sort_dir =
        request.sort_key = "<sort_key>"
        request.hosting_id =
        request.id =
```

```
request.name =
response = client.list_hosted_direct_connects(request)
print(response)
except exceptions.ClientRequestException as e:
    print(e.status_code)
    print(e.request_id)
    print(e.error_code)
    print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dc "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    // risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    // variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this
    // example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := dc.NewDcClient(
        dc.DcClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ListHostedDirectConnectsRequest{}
    limitRequest := int32(<limit>)
    request.Limit = &limitRequest
    markerRequest := "<marker>"
    request.Marker = &markerRequest
    sortKeyRequest := "<sort_key>"
    request.SortKey = &sortKeyRequest
    response, err := client.ListHostedDirectConnects(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见[API Explorer](#)的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK

错误码

请参见[错误码](#)。

5.1.6 创建托管专线连接

功能介绍

用于合作伙伴用户最终租户创建托管专线 创建者必须拥有合作伙伴资质，并且已经构建好运营(hosting)专线

调用方法

请参见[如何调用API](#)。

URI

POST /v3/{project_id}/dcaas/hosted-connects

表 5-32 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	租户项目ID

请求参数

表 5-33 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。获取Token，请参考《统一身份认证服务API参考》的“获取用户Token”章节。请求响应成功后在响应消息头中包含的“X-Subject-Token”的值即为Token值。 最小长度： 0 最大长度： 10240

表 5-34 请求 Body 参数

参数	是否必选	参数类型	描述
hosted_connect	是	CreateHostedDirectConnect object	创建托管专线参数对象

表 5-35 CreateHostedDirectConnect

参数	是否必选	参数类型	描述
name	否	String	托管物理专线的名字。 最小长度：0 最大长度：64
description	否	String	托管专线的描述信息 最小长度：0 最大长度：128
bandwidth	是	Integer	指定托管专线接入带宽,单位 Mbps 最小值：2 最大值：400000
hosting_id	是	String	hosted物理专线对应的hosting物理专线的ID
vlan	是	Integer	指定托管(hosted)专线预分配的vlan 最小值：0 最大值：3999
resource_tenant_id	是	String	为其他租户创建托管专线，指定对应的租户ID 最小长度：0 最大长度：32
peer_location	否	String	物理专线对端所在的物理位置，省/市/街道或IDC名字。 最小长度：0 最大长度：255

响应参数

状态码：201

表 5-36 响应 Body 参数

参数	参数类型	描述
request_id	String	操作请求ID
hosted_connect	HostedDirectConnect object	托管专线对象信息

表 5-37 HostedDirectConnect

参数	参数类型	描述
id	String	托管专线ID 最小长度：36 最大长度：36
tenant_id	String	实例所属项目ID。
name	String	物理专线名字 最小长度：0 最大长度：64
description	String	物理专线的描述信息 最小长度：0 最大长度：128
bandwidth	Integer	物理专线接入带宽，单位Mbps。 最小值：2 最大值：400000
location	String	专线的接入位置信息 最小长度：0 最大长度：255
peer_location	String	物理专线对端所在的物理位置，省/市/街道或IDC名字。 最小长度：0 最大长度：255
hosting_id	String	hosted物理专线对应的hosting物理专线的ID
provider	String	专线线路的提供商
admin_state_up	Boolean	管理状态：true或false 缺省值： true

参数	参数类型	描述
vlan	Integer	hosted物理专线预分配的vlan。 最小值：0 最大值：3999
status	String	操作状态，合法值是： BUILD：已开通 ACTIVE：虚拟网关正常 DOWN：专线对应的端口处于down的状态，可能在线路故障等异常。 ERROR：虚拟网关异常 PENDING_DELETE：删除中 PENDING_UPDATE：更新中 PENDING_CREATE：创建中 枚举值： <ul style="list-style-type: none">● BUILD● ACTIVE● DOWN● ERROR● PENDING_DELETE● PENDING_UPDATE● PENDING_CREATE
apply_time	String	物理专线申请时间。采用UTC时间格式，格式为： yyyy-MM-ddTHH:mm:ss.SSSZ
create_time	String	物理专线创建时间。采用UTC时间格式，格式为： yyyy-MM-ddTHH:mm:ss.SSSZ
provider_statuses	String	物理专线的运营商操作状态，合法值是： ACTIVE， DOWN 枚举值： <ul style="list-style-type: none">● ACTIVE● DOWN
port_type	String	物理专线接入接口的类型，支持1G 10G 40G 100G 枚举值： <ul style="list-style-type: none">● 1G● 10G● 40G● 100G

参数	参数类型	描述
type	String	物理专线的类型，类型包括标准(standard)，运营专线(hosting)，托管专线 (hosted)。 缺省值： standard 枚举值： <ul style="list-style-type: none">● standard● hosting● hosted● onestop_standard● onestop_hosted

请求示例

创建一个托管物理连接，设置带宽为10，VLAN为441。

```
POST https://{dc_endpoint}/v3/6fbc9263116a4b68818cf1edce16bc4f/dcaas/hosted-connects
```

```
{
  "hosted_connect" : {
    "name" : "client-dc-faf1",
    "description" : "Hosted Connect",
    "resource_tenant_id" : "0605768a3300d5762f82c01180692873",
    "hosting_id" : "2cfb53be-b05f-40d5-a2f8-3a59ac383836",
    "vlan" : 441,
    "bandwidth" : 10
  }
}
```

响应示例

状态码： 201

Created

- 成功创建托管专线连接

```
{
  "hosted_connect" : {
    "id" : "0278b472-ffa5-4eb3-8c0d-979d479f8ef6",
    "name" : "client-dc-faf1",
    "description" : "Hosted Connect",
    "tenant_id" : "0605768a3300d5762f82c01180692873",
    "hosting_id" : "2cfb53be-b05f-40d5-a2f8-3a59ac383836",
    "vlan" : 441,
    "bandwidth" : 10,
    "location" : "ExampleLocation",
    "peer_location" : "",
    "provider" : "ExampleProvider",
    "type" : "hosted",
    "port_type" : "10G",
    "provider_status" : "ACTIVE",
    "status" : "ACTIVE",
    "apply_time" : "2022-07-13T08:25:38.000Z",
    "admin_state_up" : true,
    "create_time" : "2022-07-13T08:25:38.000Z"
  },
  "request_id" : "a59a3776faa1d055f8124dc7b0977a90"
}
```

SDK 代码示例

SDK代码示例如下。

Java

创建一个托管物理连接，设置带宽为10，VLAN为441。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.dc.v3.region.DcRegion;
import com.huaweicloud.sdk.dc.v3.*;
import com.huaweicloud.sdk.dc.v3.model.*;

public class CreateHostedDirectConnectSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
        // security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
        // environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running
        // this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
            .withAk(ak)
            .withSk(sk);

        DcClient client = DcClient.newBuilder()
            .withCredential(auth)
            .withRegion(DcRegion.valueOf("<YOUR REGION>"))
            .build();
        CreateHostedDirectConnectRequest request = new CreateHostedDirectConnectRequest();
        CreateHostedDirectConnectRequestBody body = new CreateHostedDirectConnectRequestBody();
        CreateHostedDirectConnect hostedConnectbody = new CreateHostedDirectConnect();
        hostedConnectbody.withName("client-dc-faf1")
            .withDescription("Hosted Connect")
            .withBandwidth(10)
            .withHostingId("2cfb53be-b05f-40d5-a2f8-3a59ac383836")
            .withVlan(441)
            .withResourceTenantId("0605768a3300d5762f82c01180692873");
        body.withHostedConnect(hostedConnectbody);
        request.withBody(body);
        try {
            CreateHostedDirectConnectResponse response = client.createHostedDirectConnect(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

创建一个托管物理连接，设置带宽为10，VLAN为441。

```
# coding: utf-8

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdc.v3.region.dc_region import DcRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdc.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    # risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    # variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this
    # example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.getenv("CLOUD_SDK_AK")
    sk = os.getenv("CLOUD_SDK_SK")

    credentials = BasicCredentials(ak, sk) \

    client = DcClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DcRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = CreateHostedDirectConnectRequest()
        hostedConnectbody = CreateHostedDirectConnect(
            name="client-dc-faf1",
            description="Hosted Connect",
            bandwidth=10,
            hosting_id="2cfb53be-b05f-40d5-a2f8-3a59ac383836",
            vlan=441,
            resource_tenant_id="0605768a3300d5762f82c01180692873"
        )
        request.body = CreateHostedDirectConnectRequestBody(
            hosted_connect=hostedConnectbody
        )
        response = client.create_hosted_direct_connect(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

创建一个托管物理连接，设置带宽为10，VLAN为441。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dc "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    // risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    // variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this
    // example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
```

```
ak := os.Getenv("CLOUD_SDK_AK")
sk := os.Getenv("CLOUD_SDK_SK")

auth := basic.NewCredentialsBuilder().
    WithAk(ak).
    WithSk(sk).
    Build()

client := dc.NewDcClient(
    dc.DcClientBuilder().
        WithRegion(region.ValueOf("<YOUR REGION>")).
        WithCredential(auth).
        Build())

request := &model.CreateHostedDirectConnectRequest{
    nameHostedConnect:= "client-dc-faf1"
    descriptionHostedConnect:= "Hosted Connect"
    hostedConnectbody := &model.CreateHostedDirectConnect{
        Name: &nameHostedConnect,
        Description: &descriptionHostedConnect,
        Bandwidth: int32(10),
        HostingId: "2cfb53be-b05f-40d5-a2f8-3a59ac383836",
        Vlan: int32(441),
        ResourceTenantId: "0605768a3300d5762f82c01180692873",
    }
}
request.Body = &model.CreateHostedDirectConnectRequestBody{
    HostedConnect: hostedConnectbody,
}
response, err := client.CreateHostedDirectConnect(request)
if err == nil {
    fmt.Printf("%+v\n", response)
} else {
    fmt.Println(err)
}
}
```

更多

更多编程语言的SDK代码示例，请参见[API Explorer](#)的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
201	Created

错误码

请参见[错误码](#)。

5.1.7 查询租户的托管专线详情

功能介绍

查询合法合作伙伴的Hosted专线类型。

调用方法

请参见[如何调用API](#)。

URI

GET /v3/{project_id}/dcaas/hosted-connects/{hosted_connect_id}

表 5-38 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	租户项目ID
hosted_connect_id	是	String	托管专线连接ID。 最小长度：36 最大长度：36

表 5-39 Query 参数

参数	是否必选	参数类型	描述
limit	否	Integer	每页返回的个数。取值范围：1~2000。 最小值：1 最大值：2000 缺省值：2000
marker	否	String	上一页最后一条资源记录的ID，为空时为查询第一页。使用说明：必须与limit一起使用。 最小长度：0 最大长度：36
fields	否	Array	显示字段列表 数组长度：1 - 5
sort_dir	否	Array	返回结果按照升序(asc)或降序(desc)排列，默认为asc
sort_key	否	String	排序字段。 缺省值：id 最小长度：0 最大长度：36
hosting_id	否	Array	根据运营专线ID过滤托管专线列表 数组长度：0 - 5

请求参数

表 5-40 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。获取Token，请参考《统一身份认证服务API参考》的“获取用户Token”章节。请求响应成功后在响应消息头中包含的“X-Subject-Token”的值即为Token值。 最小长度：0 最大长度：10240

响应参数

状态码：200

表 5-41 响应 Body 参数

参数	参数类型	描述
request_id	String	操作请求ID
hosted_connect	HostedDirectConnect object	托管专线对象信息

表 5-42 HostedDirectConnect

参数	参数类型	描述
id	String	托管专线ID 最小长度：36 最大长度：36
tenant_id	String	实例所属项目ID。
name	String	物理专线名字 最小长度：0 最大长度：64
description	String	物理专线的描述信息 最小长度：0 最大长度：128

参数	参数类型	描述
bandwidth	Integer	物理专线接入带宽，单位Mbps。 最小值：2 最大值：400000
location	String	专线的接入位置信息 最小长度：0 最大长度：255
peer_location	String	物理专线对端所在的物理位置，省/市/街道或IDC名字。 最小长度：0 最大长度：255
hosting_id	String	hosted物理专线对应的hosting物理专线的ID
provider	String	专线线路的提供商
admin_state_up	Boolean	管理状态：true或false 缺省值：true
vlan	Integer	hosted物理专线预分配的vlan。 最小值：0 最大值：3999
status	String	操作状态，合法值是：BUILD：已开通 ACTIVE：虚拟网关正常 DOWN：专线对应的端口处于down的状态，可能存在线路故障等异常。 ERROR：虚拟网关异常 PENDING_DELETE：删除中 PENDING_UPDATE：更新中 PENDING_CREATE：创建中 枚举值： <ul style="list-style-type: none">● BUILD● ACTIVE● DOWN● ERROR● PENDING_DELETE● PENDING_UPDATE● PENDING_CREATE
apply_time	String	物理专线申请时间。采用UTC时间格式，格式为：yyyy-MM-ddTHH:mm:ss.SSSZ
create_time	String	物理专线创建时间。采用UTC时间格式，格式为：yyyy-MM-ddTHH:mm:ss.SSSZ

参数	参数类型	描述
provider_statuses	String	物理专线的运营商操作状态，合法值是： ACTIVE， DOWN 枚举值： <ul style="list-style-type: none">● ACTIVE● DOWN
port_type	String	物理专线接入接口的类型，支持1G 10G 40G 100G 枚举值： <ul style="list-style-type: none">● 1G● 10G● 40G● 100G
type	String	物理专线的类型，类型包括标准(standard)，运营专线(hosting)，托管专线 (hosted)。 缺省值： standard 枚举值： <ul style="list-style-type: none">● standard● hosting● hosted● onestop_standard● onestop_hosted

请求示例

查询合作伙伴的托管专线详情

```
GET https://{dc_endpoint}/v3/6fbe9263116a4b68818cf1edce16bc4f/dcaas/hosted-connects/0278b472-ffa5-4eb3-8c0d-979d479f8ef6
```

响应示例

状态码： 200

OK

- 成功查询托管专线连接详情信息

```
{  
  "hosted_connect": {  
    "id": "0278b472-ffa5-4eb3-8c0d-979d479f8ef6",  
    "name": "client-dc-faf1",  
    "description": "",  
    "tenant_id": "0605768a3300d5762f82c01180692873",  
    "hosting_id": "2cfb53be-b05f-40d5-a2f8-3a59ac383836",  
    "vlan": 441,  
    "bandwidth": 10,  
    "location": "ExampleLocation",  
    "peer_location": ""  
  }  
}
```

```
"provider" : "ExampleProvider",
"type" : "hosted",
"port_type" : "10G",
"provider_status" : "ACTIVE",
"status" : "ACTIVE",
"apply_time" : "2022-07-13T08:25:38.000Z",
"admin_state_up" : true,
"create_time" : "2022-07-13T08:25:38.000Z"
},
"request_id" : "a59a3776faa1d055f8124dc7b0977a90"
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.dc.v3.region.DcRegion;
import com.huaweicloud.sdk.dc.v3.*;
import com.huaweicloud.sdk.dc.v3.model.*;

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

public class ShowHostedDirectConnectSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
        // security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
        // environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running
        // this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
            .withAk(ak)
            .withSk(sk);

        DcClient client = DcClient.newBuilder()
            .withCredential(auth)
            .withRegion(DcRegion.valueOf("<YOUR REGION>"))
            .build();
        ShowHostedDirectConnectRequest request = new ShowHostedDirectConnectRequest();
        request.withLimit(<limit>);
        request.withMarker("<marker>");
        request.withFields();
        request.withSortDir();
        request.withSortKey("<sort_key>");
        request.withHostingId();
        try {
            ShowHostedDirectConnectResponse response = client.showHostedDirectConnect(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
        }
    }
}
```

```
        System.out.println(e.getRequestId());
        System.out.println(e.getErrorCode());
        System.out.println(e.getErrorMsg());
    }
}
}
```

Python

```
# coding: utf-8

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdc.v3.region.dc_region import DcRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdc.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    # risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    # variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this
    # example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.getenv("CLOUD_SDK_AK")
    sk = os.getenv("CLOUD_SDK_SK")

    credentials = BasicCredentials(ak, sk) \

    client = DcClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DcRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowHostedDirectConnectRequest()
        request.limit = <limit>
        request.marker = "<marker>"
        request.fields =
        request.sort_dir =
        request.sort_key = "<sort_key>"
        request.hosting_id =
        response = client.show_hosted_direct_connect(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dc "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    // risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    // variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this
    // example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
```

```
auth := basic.NewCredentialsBuilder().
    WithAk(ak).
    WithSk(sk).
    Build()

client := dc.NewDcClient(
    dc.DcClientBuilder().
        WithRegion(region.ValueOf("<YOUR REGION>")).
        WithCredential(auth).
        Build())

request := &model.ShowHostedDirectConnectRequest{}
limitRequest:= int32(<limit>)
request.Limit = &limitRequest
markerRequest:= "<marker>"
request.Marker = &markerRequest
sortKeyRequest:= "<sort_key>"
request.SortKey = &sortKeyRequest
response, err := client.ShowHostedDirectConnect(request)
if err == nil {
    fmt.Printf("%+v\n", response)
} else {
    fmt.Println(err)
}
}
```

更多

更多编程语言的SDK代码示例，请参见[API Explorer](#)的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK

错误码

请参见[错误码](#)。

5.1.8 更新托管专线连接

功能介绍

合作伙伴更新托管专线。

调用方法

请参见[如何调用API](#)。

URI

PUT /v3/{project_id}/dcaas/hosted-connects/{hosted_connect_id}

表 5-43 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	租户项目ID
hosted_connect_id	是	String	托管专线连接ID。 最小长度：36 最大长度：36

请求参数

表 5-44 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。获取Token，请参考《统一身份认证服务API参考》的“获取用户Token”章节。请求响应成功后在响应消息头中包含的“X-Subject-Token”的值即为Token值。 最小长度：0 最大长度：10240

表 5-45 请求 Body 参数

参数	是否必选	参数类型	描述
hosted_connect	否	UpdateHostedDirectConnect object	更新托管专线对象

表 5-46 UpdateHostedDirectConnect

参数	是否必选	参数类型	描述
name	否	String	物理专线的名字 最小长度：0 最大长度：64
description	否	String	物理专线的描述信息 最小长度：0 最大长度：128

参数	是否必选	参数类型	描述
bandwidth	否	Integer	指定托管专线接入带宽,单位Mbps 最小值: 2 最大值: 400000
peer_location	否	String	物理专线对端所在的物理位置,省/市/街道或IDC名字 最小长度: 0 最大长度: 255

响应参数

状态码: 200

表 5-47 响应 Body 参数

参数	参数类型	描述
request_id	String	操作请求ID
hosted_connect	HostedDirectConnect object	托管专线对象信息

表 5-48 HostedDirectConnect

参数	参数类型	描述
id	String	托管专线ID 最小长度: 36 最大长度: 36
tenant_id	String	实例所属项目ID。
name	String	物理专线名字 最小长度: 0 最大长度: 64
description	String	物理专线的描述信息 最小长度: 0 最大长度: 128
bandwidth	Integer	物理专线接入带宽, 单位Mbps。 最小值: 2 最大值: 400000

参数	参数类型	描述
location	String	专线的接入位置信息 最小长度：0 最大长度：255
peer_location	String	物理专线对端所在的物理位置，省/市/街道或IDC名字。 最小长度：0 最大长度：255
hosting_id	String	hosted物理专线对应的hosting物理专线的ID
provider	String	专线线路的提供商
admin_state_up	Boolean	管理状态：true或false 缺省值： true
vlan	Integer	hosted物理专线预分配的vlan。 最小值：0 最大值： 3999
status	String	操作状态，合法值是：BUILD：已开通 ACTIVE：虚拟网关正常 DOWN：专线对应的端口处于down的状态，可能存在线路故障等异常。 ERROR：虚拟网关异常 PENDING_DELETE：删除中 PENDING_UPDATE：更新中 PENDING_CREATE：创建中 枚举值： <ul style="list-style-type: none">● BUILD● ACTIVE● DOWN● ERROR● PENDING_DELETE● PENDING_UPDATE● PENDING_CREATE
apply_time	String	物理专线申请时间。采用UTC时间格式，格式为：yyyy-MM-ddTHH:mm:ss.SSSZ
create_time	String	物理专线创建时间。采用UTC时间格式，格式为：yyyy-MM-ddTHH:mm:ss.SSSZ
provider_statuses	String	物理专线的运营商操作状态，合法值是：ACTIVE，DOWN 枚举值： <ul style="list-style-type: none">● ACTIVE● DOWN

参数	参数类型	描述
port_type	String	物理专线接入接口的类型，支持1G 10G 40G 100G 枚举值： <ul style="list-style-type: none">● 1G● 10G● 40G● 100G
type	String	物理专线的类型，类型包括标准(standard)，运营专线(hosting)，托管专线 (hosted)。 缺省值： standard 枚举值： <ul style="list-style-type: none">● standard● hosting● hosted● onestop_standard● onestop_hosted

请求示例

更新托管专线连接的名称和描述信息

```
PUT https://{dc_endpoint}/v3/6f9e9263116a4b68818cf1edce16bc4f/dcaas/hosted-connects/0278b472-ffa5-4eb3-8c0d-979d479f8ef6
```

```
{
  "hosted_connect" : {
    "name" : "client-dc-faf1",
    "description" : ""
  }
}
```

响应示例

状态码： 200

OK

- 成功更新托管专线连接

```
{
  "hosted_connect" : {
    "id" : "0278b472-ffa5-4eb3-8c0d-979d479f8ef6",
    "name" : "client-dc-faf1",
    "description" : "",
    "tenant_id" : "0605768a3300d5762f82c01180692873",
    "hosting_id" : "2cfb53be-b05f-40d5-a2f8-3a59ac383836",
    "vlan" : 441,
    "bandwidth" : 10,
    "location" : "ExampleLocation",
    "peer_location" : "",
    "provider" : "ExampleProvider",
  }
}
```

```
"type": "hosted",
"port_type": "10G",
"provider_status": "ACTIVE",
"status": "ACTIVE",
"apply_time": "2022-07-13T08:25:38.000Z",
"admin_state_up": true,
"create_time": "2022-07-13T08:25:38.000Z"
},
"request_id": "a59a3776faa1d055f8124dc7b0977a90"
}
```

SDK 代码示例

SDK代码示例如下。

Java

更新托管专线连接的名称和描述信息

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.dc.v3.region.DcRegion;
import com.huaweicloud.sdk.dc.v3.*;
import com.huaweicloud.sdk.dc.v3.model.*;

public class UpdateHostedDirectConnectSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
        // security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
        // environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running
        // this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
            .withAk(ak)
            .withSk(sk);

        DcClient client = DcClient.newBuilder()
            .withCredential(auth)
            .withRegion(DcRegion.valueOf("<YOUR REGION>"))
            .build();
        UpdateHostedDirectConnectRequest request = new UpdateHostedDirectConnectRequest();
        UpdateHostedDirectConnectRequestBody body = new UpdateHostedDirectConnectRequestBody();
        UpdateHostedDirectConnect hostedConnectbody = new UpdateHostedDirectConnect();
        hostedConnectbody.setName("client-dc-faf1")
            .withDescription("");
        body.withHostedConnect(hostedConnectbody);
        request.withBody(body);
        try {
            UpdateHostedDirectConnectResponse response = client.updateHostedDirectConnect(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
        }
    }
}
```

```
        System.out.println(e.getRequestId());
        System.out.println(e.getErrorCode());
        System.out.println(e.getErrorMsg());
    }
}
}
```

Python

更新托管专线连接的名称和描述信息

```
# coding: utf-8

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdc.v3.region.dc_region import DcRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdc.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    # risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    # variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this
    # example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.getenv("CLOUD_SDK_AK")
    sk = os.getenv("CLOUD_SDK_SK")

    credentials = BasicCredentials(ak, sk) \

    client = DcClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DcRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = UpdateHostedDirectConnectRequest()
        hostedConnectbody = UpdateHostedDirectConnect(
            name="client-dc-faf1",
            description=""
        )
        request.body = UpdateHostedDirectConnectRequestBody(
            hosted_connect=hostedConnectbody
        )
        response = client.update_hosted_direct_connect(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

更新托管专线连接的名称和描述信息

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dc "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    // risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
```

```
variables and decrypted during use to ensure security.
// In this example, AK and SK are stored in environment variables for authentication. Before running this
example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
ak := os.Getenv("CLOUD_SDK_AK")
sk := os.Getenv("CLOUD_SDK_SK")

auth := basic.NewCredentialsBuilder().
    WithAk(ak).
    WithSk(sk).
    Build()

client := dc.NewDcClient(
    dc.DcClientBuilder().
        WithRegion(region.ValueOf("<YOUR REGION>")).
        WithCredential(auth).
        Build())

request := &model.UpdateHostedDirectConnectRequest{
    nameHostedConnect:= "client-dc-faf1"
    descriptionHostedConnect:= ""
    hostedConnectbody := &model.UpdateHostedDirectConnect{
        Name: &nameHostedConnect,
        Description: &descriptionHostedConnect,
    }
}
request.Body = &model.UpdateHostedDirectConnectRequestBody{
    HostedConnect: hostedConnectbody,
}
response, err := client.UpdateHostedDirectConnect(request)
if err == nil {
    fmt.Printf("%+v\n", response)
} else {
    fmt.Println(err)
}
}
```

更多

更多编程语言的SDK代码示例，请参见[API Explorer](#)的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK

错误码

请参见[错误码](#)。

5.1.9 删除托管专线连接

功能介绍

合作伙伴删除托管专线

调用方法

请参见[如何调用API](#)。

URI

DELETE /v3/{project_id}/dcaas/hosted-connects/{hosted_connect_id}

表 5-49 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	租户项目ID
hosted_connect_id	是	String	托管专线连接ID。 最小长度： 36 最大长度： 36

请求参数

表 5-50 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。获取Token，请参考《统一身份认证服务API参考》的“获取用户Token”章节。请求响应成功后在响应消息头中包含的“X-Subject-Token”的值即为Token值。 最小长度： 0 最大长度： 10240

响应参数

无

请求示例

删除托管专线连接

```
DELETE https://{dc_endpoint}/v3/08d5a9564a704afda6039ae2babbe3c/dcaas/hosted-connects/94c2b814-99dc-939a-e811-ae84c61ea3ff
```

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.dc.v3.region.DcRegion;
import com.huaweicloud.sdk.dc.v3.*;
import com.huaweicloud.sdk.dc.v3.model.*;

public class DeleteHostedDirectConnectSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
        // security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
        // environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running
        // this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
            .withAk(ak)
            .withSk(sk);

        DcClient client = DcClient.newBuilder()
            .withCredential(auth)
            .withRegion(DcRegion.valueOf("<YOUR REGION>"))
            .build();
        DeleteHostedDirectConnectRequest request = new DeleteHostedDirectConnectRequest();
        try {
            DeleteHostedDirectConnectResponse response = client.deleteHostedDirectConnect(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdc.v3.region.dc_region import DcRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdc.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    # risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    # variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this
    # example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.getenv("CLOUD_SDK_AK")
    sk = os.getenv("CLOUD_SDK_SK")
```



```
credentials = BasicCredentials(ak, sk) \

client = DcClient.new_builder() \
    .with_credentials(credentials) \
    .with_region(DcRegion.value_of("<YOUR REGION>")) \
    .build()

try:
    request = DeleteHostedDirectConnectRequest()
    response = client.delete_hosted_direct_connect(request)
    print(response)
except exceptions.ClientRequestException as e:
    print(e.status_code)
    print(e.request_id)
    print(e.error_code)
    print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dc "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    // risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    // variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this
    // example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := dc.NewDcClient(
        dc.DcClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.DeleteHostedDirectConnectRequest{}
    response, err := client.DeleteHostedDirectConnect(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见[API Explorer](#)的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
204	No Content

错误码

请参见[错误码](#)。

5.2 虚拟网关

5.2.1 查询虚拟网关详情

功能介绍

查询指定虚拟网关的详细信息

调用方法

请参见[如何调用API](#)。

URI

GET /v3/{project_id}/dcaas/virtual-gateways/{virtual_gateway_id}

表 5-51 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	租户项目ID
virtual_gateway_id	是	String	虚拟网关ID

表 5-52 Query 参数

参数	是否必选	参数类型	描述
fields	否	Array	显示字段列表 数组长度：1 - 5

请求参数

表 5-53 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。获取Token，请参考《统一身份认证服务API参考》的“获取用户Token”章节。请求响应成功后在响应消息头中包含的“X-Subject-Token”的值即为Token值。 最小长度：0 最大长度：10240

响应参数

状态码：200

表 5-54 响应 Body 参数

参数	参数类型	描述
virtual_gateway	VirtualGateway object	虚拟网关对象
request_id	String	操作请求ID

表 5-55 VirtualGateway

参数	参数类型	描述
id	String	虚拟网关的ID
vpc_id	String	虚拟网关接入的VPC的ID
tenant_id	String	实例所属项目ID。 最小长度：32 最大长度：32
name	String	虚拟网关的名字 最小长度：0 最大长度：64
description	String	虚拟网关的描述 最小长度：0 最大长度：128

参数	参数类型	描述
type	String	虚拟网关类型: default 缺省值: default
local_ep_group	Array of strings	虚拟网关到访问云上服务IPv4子网列表, 通常是vpc的cidrs
local_ep_group_ipv6	Array of strings	预留字段用于虚拟网关到访问云上服务IPv6子网列表, 通常是vpc的cidrs
admin_state_up	Boolean	管理状态: true或false 缺省值: true
status	String	操作状态, 合法值是: ACTIVE, DOWN, BUILD, ERROR, PENDING_CREATE, PENDING_UPDATE, PENDING_DELETE
bgp_asn	Integer	虚拟网关本地的BGP自治域号(asn) 最小值: 1 最大值: 4294967295
enterprise_project_id	String	实例所属企业项目ID 最小长度: 36 最大长度: 36
device_id	String	归属的设备ID
redundant_device_id	String	归属的冗余设备ID
public_border_group	String	归属的可用区对应的边界组(public border group), 标识是否homezone局点。
tags	Array of Tag objects	标签信息 数组长度: 0 - 10

表 5-56 Tag

参数	参数类型	描述
key	String	标签键, 最大长度36个unicode字符, 格式为大小写字母, 数字, 中划线“-”, 下划线“_”, 中文。 最小长度: 0 最大长度: 36

参数	参数类型	描述
value	String	标签值，最大长度43个unicode字符，格式为大写字母，数字，中划线“-”，下划线“_”，点“.”，中文。 最小长度： 0 最大长度： 43

请求示例

查询虚拟网关详情

```
GET https://{dc_endpoint}/v3/0605768a3300d5762f82c01180692873/dcaas/virtual-gateways/  
20082c1b-3c99-48d8-8e8c-116af5d7e9f0
```

响应示例

状态码：**200**

OK

- 成功查询接入虚拟私有云的虚拟网关详情。

```
{  
  "virtual_gateway": {  
    "id": "20082c1b-3c99-48d8-8e8c-116af5d7e9f0",  
    "name": "vgw-c7b22",  
    "description": "",  
    "tenant_id": "0605768a3300d5762f82c01180692873",  
    "vpc_id": "6592c28e-95d7-4b0a-9f61-004fdf03420c",  
    "device_id": "26.151.63.100",  
    "redundant_device_id": "26.152.128.20",  
    "type": "default",  
    "status": "ACTIVE",  
    "admin_state_up": true,  
    "bgp_asn": 64512,  
    "local_ep_group": [ "192.168.1.0/24" ],  
    "enterprise_project_id": "0",  
    "public_border_group": "center"  
  },  
  "request_id": "765f7aaf8f2edd0e719de564ef72e2de"  
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;  
  
import com.huaweicloud.sdk.core.auth.ICredential;  
import com.huaweicloud.sdk.core.auth.BasicCredentials;  
import com.huaweicloud.sdk.core.exception.ConnectionException;  
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;  
import com.huaweicloud.sdk.core.exception.ServiceResponseException;  
import com.huaweicloud.sdk.dc.v3.region.DcRegion;  
import com.huaweicloud.sdk.dc.v3.*;  
import com.huaweicloud.sdk.dc.v3.model.*;  
  
import java.util.List;
```

```
import java.util.ArrayList;

public class ShowVirtualGatewaySolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
        security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
        environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running
        this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
            .withAk(ak)
            .withSk(sk);

        DcClient client = DcClient.newBuilder()
            .withCredential(auth)
            .withRegion(DcRegion.valueOf("<YOUR REGION>"))
            .build();
        ShowVirtualGatewayRequest request = new ShowVirtualGatewayRequest();
        request.withFields();
        try {
            ShowVirtualGatewayResponse response = client.showVirtualGateway(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdc.v3.region.dc_region import DcRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdc.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this
    example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.getenv("CLOUD_SDK_AK")
    sk = os.getenv("CLOUD_SDK_SK")

    credentials = BasicCredentials(ak, sk) \

    client = DcClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DcRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowVirtualGatewayRequest()
        request.fields =
        response = client.show_virtual_gateway(request)
```

```
print(response)
except exceptions.ClientRequestException as e:
    print(e.status_code)
    print(e.request_id)
    print(e.error_code)
    print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dc "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    // risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    // variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this
    // example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := dc.NewDcClient(
        dc.DcClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowVirtualGatewayRequest{}
    response, err := client.ShowVirtualGateway(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见[API Explorer](#)的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK

错误码

请参见[错误码](#)。

5.2.2 更新虚拟网关信息

功能介绍

更新虚拟网关的信息

调用方法

请参见[如何调用API](#)。

URI

PUT /v3/{project_id}/dcaas/virtual-gateways/{virtual_gateway_id}

表 5-57 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	租户项目ID
virtual_gateway_id	是	String	虚拟网关ID

请求参数

表 5-58 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。获取Token，请参考《统一身份认证服务API参考》的“获取用户Token”章节。请求响应成功后在响应消息头中包含的“X-Subject-Token”的值即为Token值。 最小长度：0 最大长度：10240

表 5-59 请求 Body 参数

参数	是否必选	参数类型	描述
virtual_gateway	否	UpdateVirtualGateway object	更新虚拟网关参数

表 5-60 UpdateVirtualGateway

参数	是否必选	参数类型	描述
name	否	String	更新虚拟网关的名字 最小长度：0 最大长度：64
description	否	String	虚拟网关的描述信息 最小长度：0 最大长度：128
local_ep_group	否	Array of strings	虚拟网关到访问云上服务IPv4子网列表，通常是vpc的cidrs 数组长度：1 - 200
local_ep_group_ipv6	否	Array of strings	虚拟网关到访问云上服务IPv6子网列表，通常是vpc的cidrs。 数组长度：1 - 50

响应参数

状态码：200

表 5-61 响应 Body 参数

参数	参数类型	描述
virtual_gateway	VirtualGateway object	虚拟网关对象
request_id	String	操作请求ID

表 5-62 VirtualGateway

参数	参数类型	描述
id	String	虚拟网关的ID
vpc_id	String	虚拟网关接入的VPC的ID
tenant_id	String	实例所属项目ID。 最小长度：32 最大长度：32
name	String	虚拟网关的名字 最小长度：0 最大长度：64

参数	参数类型	描述
description	String	虚拟网关的描述 最小长度：0 最大长度：128
type	String	虚拟网关类型：default 缺省值：default
local_ep_group	Array of strings	虚拟网关到访问云上服务IPv4子网列表，通常是vpc的cidrs
local_ep_group_ipv6	Array of strings	预留字段用于虚拟网关到访问云上服务IPv6子网列表，通常是vpc的cidrs
admin_state_up	Boolean	管理状态：true或false 缺省值：true
status	String	操作状态，合法值是：ACTIVE, DOWN, BUILD, ERROR, PENDING_CREATE, PENDING_UPDATE, PENDING_DELETE
bgp_asn	Integer	虚拟网关本地的BGP自治域号(asn) 最小值：1 最大值：4294967295
enterprise_project_id	String	实例所属企业项目ID 最小长度：36 最大长度：36
device_id	String	归属的设备ID
redundant_device_id	String	归属的冗余设备ID
public_border_group	String	归属的可用区对应的边界组(public border group)，标识是否homezone局点。
tags	Array of Tag objects	标签信息 数组长度：0 - 10

表 5-63 Tag

参数	参数类型	描述
key	String	标签键，最大长度36个unicode字符，格式为大小写字母，数字，中划线“-”，下划线“_”，中文。 最小长度：0 最大长度：36

参数	参数类型	描述
value	String	标签值，最大长度43个unicode字符，格式为大写字母，数字，中划线“-”，下划线“_”，点“.”，中文。 最小长度： 0 最大长度： 43

请求示例

更新虚拟网关名称和描述，将虚拟网关访问云上服务IPv4子网列表修改为192.168.3.0/24。

```
PUT https://{dc_endpoint}/v3/0605768a3300d5762f82c01180692873/dcaas/virtual-gateways/  
20082c1b-3c99-48d8-8e8c-116af5d7e9f0  
  
{  
  "virtual_gateway": {  
    "name": "update-vgw-c7b22",  
    "description": "",  
    "local_ep_group": [ "192.168.3.0/24" ]  
  }  
}
```

响应示例

状态码： 200

OK

- 成功更新接入虚拟私有云的虚拟网关信息。

```
{  
  "virtual_gateway": {  
    "id": "20082c1b-3c99-48d8-8e8c-116af5d7e9f0",  
    "name": "update-vgw-c7b22",  
    "description": "",  
    "tenant_id": "0605768a3300d5762f82c01180692873",  
    "vpc_id": "6592c28e-95d7-4b0a-9f61-004fdf03420c",  
    "device_id": "26.151.63.100",  
    "redundant_device_id": "26.152.128.20",  
    "type": "default",  
    "status": "ACTIVE",  
    "admin_state_up": true,  
    "bgp_asn": 64512,  
    "local_ep_group": [ "192.168.3.0/24" ],  
    "enterprise_project_id": "0",  
    "public_border_group": "center"  
  },  
  "request_id": "765f7aaf8f2edd0e719de564ef72e2de"  
}
```

SDK 代码示例

SDK代码示例如下。

Java

更新虚拟网关名称和描述，将虚拟网关访问云上服务IPv4子网列表修改为192.168.3.0/24。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.dc.v3.region.DcRegion;
import com.huaweicloud.sdk.dc.v3.*;
import com.huaweicloud.sdk.dc.v3.model.*;

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

public class UpdateVirtualGatewaySolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
        // security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
        // environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running
        // this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
            .withAk(ak)
            .withSk(sk);

        DcClient client = DcClient.newBuilder()
            .withCredential(auth)
            .withRegion(DcRegion.valueOf("<YOUR REGION>"))
            .build();

        UpdateVirtualGatewayRequest request = new UpdateVirtualGatewayRequest();
        UpdateVirtualGatewayRequestBody body = new UpdateVirtualGatewayRequestBody();
        List<String> listVirtualGatewayLocalEpGroup = new ArrayList<>();
        listVirtualGatewayLocalEpGroup.add("192.168.3.0/24");
        UpdateVirtualGateway virtualGatewaybody = new UpdateVirtualGateway();
        virtualGatewaybody.withName("update-vgw-c7b22")
            .withDescription("")
            .withLocalEpGroup(listVirtualGatewayLocalEpGroup);
        body.withVirtualGateway(virtualGatewaybody);
        request.withBody(body);
        try {
            UpdateVirtualGatewayResponse response = client.updateVirtualGateway(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

更新虚拟网关名称和描述，将虚拟网关访问云上服务IPv4子网列表修改为192.168.3.0/24。

```
# coding: utf-8

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdcdc.v3.region.dc_region import DcRegion
```

```
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdcdc.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this
    example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.getenv("CLOUD_SDK_AK")
    sk = os.getenv("CLOUD_SDK_SK")

    credentials = BasicCredentials(ak, sk) \

    client = DcClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DcRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = UpdateVirtualGatewayRequest()
        listLocalEpGroupVirtualGateway = [
            "192.168.3.0/24"
        ]
        virtualGatewaybody = UpdateVirtualGateway(
            name="update-vgw-c7b22",
            description="",
            local_ep_group=listLocalEpGroupVirtualGateway
        )
        request.body = UpdateVirtualGatewayRequestBody(
            virtual_gateway=virtualGatewaybody
        )
        response = client.update_virtual_gateway(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

更新虚拟网关名称和描述，将虚拟网关访问云上服务IPv4子网列表修改为192.168.3.0/24。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dc "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this
    example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()
```

```
client := dc.NewDcClient(
    dc.DcClientBuilder().
        WithRegion(region.ValueOf("<YOUR REGION>")).
        WithCredential(auth).
        Build())

request := &model.UpdateVirtualGatewayRequest{}
var listLocalEpGroupVirtualGateway = []string{
    "192.168.3.0/24",
}
nameVirtualGateway:= "update-vgw-c7b22"
descriptionVirtualGateway:= ""
virtualGatewaybody := &model.UpdateVirtualGateway{
    Name: &nameVirtualGateway,
    Description: &descriptionVirtualGateway,
    LocalEpGroup: &listLocalEpGroupVirtualGateway,
}
request.Body = &model.UpdateVirtualGatewayRequestBody{
    VirtualGateway: virtualGatewaybody,
}
response, err := client.UpdateVirtualGateway(request)
if err == nil {
    fmt.Printf("%+v\n", response)
} else {
    fmt.Println(err)
}
```

更多

更多编程语言的SDK代码示例，请参见[API Explorer](#)的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK

错误码

请参见[错误码](#)。

5.2.3 删除虚拟网关

功能介绍

删除指定的虚拟网关

调用方法

请参见[如何调用API](#)。

URI

DELETE /v3/{project_id}/dcaas/virtual-gateways/{virtual_gateway_id}

表 5-64 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	租户项目ID
virtual_gateway_id	是	String	虚拟网关ID

请求参数

表 5-65 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。获取Token，请参考《统一身份认证服务API参考》的“获取用户Token”章节。请求响应成功后在响应消息头中包含的“X-Subject-Token”的值即为Token值。 最小长度： 0 最大长度： 10240

响应参数

无

请求示例

删除虚拟网关

```
DELETE https://{dc_endpoint}/v3/08d5a9564a704afda6039ae2babbe3c/dcaas/virtual-gateways/20082c1b-3c99-48d8-8e8c-116af5d7e9f0
```

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
```

```
import com.huaweicloud.sdk.dc.v3.region.DcRegion;
import com.huaweicloud.sdk.dc.v3.*;
import com.huaweicloud.sdk.dc.v3.model.*;

public class DeleteVirtualGatewaySolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
        // security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
        // environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running
        // this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
            .withAk(ak)
            .withSk(sk);

        DcClient client = DcClient.newBuilder()
            .withCredential(auth)
            .withRegion(DcRegion.valueOf("<YOUR REGION>"))
            .build();
        DeleteVirtualGatewayRequest request = new DeleteVirtualGatewayRequest();
        try {
            DeleteVirtualGatewayResponse response = client.deleteVirtualGateway(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdc.v3.region.dc_region import DcRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdc.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    # risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    # variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this
    # example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.getenv("CLOUD_SDK_AK")
    sk = os.getenv("CLOUD_SDK_SK")

    credentials = BasicCredentials(ak, sk) \

    client = DcClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DcRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = DeleteVirtualGatewayRequest()
```



```
response = client.delete_virtual_gateway(request)
print(response)
except exceptions.ClientRequestException as e:
    print(e.status_code)
    print(e.request_id)
    print(e.error_code)
    print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dc "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    // risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    // variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this
    // example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := dc.NewDcClient(
        dc.DcClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.DeleteVirtualGatewayRequest{}
    response, err := client.DeleteVirtualGateway(request)
    if err == nil {
        fmt.Printf("%v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见[API Explorer](#)的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
204	No Content

错误码

请参见[错误码](#)。

5.2.4 查询虚拟网关列表

功能介绍

查询虚拟网关列表

调用方法

请参见[如何调用API](#)。

URI

GET /v3/{project_id}/dcaas/virtual-gateways

表 5-66 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	租户项目ID

表 5-67 Query 参数

参数	是否必选	参数类型	描述
limit	否	Integer	每页返回的个数。取值范围： 1~2000。 最小值： 1 最大值： 2000 缺省值： 2000
marker	否	String	上一页最后一条资源记录的ID， 为空时为查询第一页。使用说明： 必须与limit一起使用。 最小长度： 0 最大长度： 36
fields	否	Array	显示字段列表 数组长度： 1 - 5
sort_dir	否	Array	返回结果按照升序(asc)或降序 (desc)排列，默认为asc
sort_key	否	String	排序字段。 缺省值： id 最小长度： 0 最大长度： 36
id	否	Array	根据资源ID过滤实例 数组长度： 1 - 5

参数	是否必选	参数类型	描述
enterprise_project_id	否	Array	根据企业项目ID过滤资源实例 数组长度：1 - 10
vpc_id	否	Array	通过VPC-ID过滤虚拟网关实例

请求参数

表 5-68 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。获取Token，请参考《统一身份认证服务API参考》的“获取用户Token”章节。请求响应成功后在响应消息头中包含的“X-Subject-Token”的值即为Token值。 最小长度：0 最大长度：10240

响应参数

状态码：200

表 5-69 响应 Body 参数

参数	参数类型	描述
request_id	String	操作请求ID
virtual_gateways	Array of VirtualGateway objects	虚拟网关对象列表
page_info	PageInfo object	分页查询页的信息

表 5-70 VirtualGateway

参数	参数类型	描述
id	String	虚拟网关的ID
vpc_id	String	虚拟网关接入的VPC的ID

参数	参数类型	描述
tenant_id	String	实例所属项目ID。 最小长度：32 最大长度：32
name	String	虚拟网关的名字 最小长度：0 最大长度：64
description	String	虚拟网关的描述 最小长度：0 最大长度：128
type	String	虚拟网关类型：default 缺省值：default
local_ep_group	Array of strings	虚拟网关到访问云上服务IPv4子网列表，通常是vpc的cidrs
local_ep_group_ipv6	Array of strings	预留字段用于虚拟网关到访问云上服务IPv6子网列表，通常是vpc的cidrs
admin_state_up	Boolean	管理状态：true或false 缺省值：true
status	String	操作状态，合法值是：ACTIVE, DOWN, BUILD, ERROR, PENDING_CREATE, PENDING_UPDATE, PENDING_DELETE
bgp_asn	Integer	虚拟网关本地的BGP自治域号(asn) 最小值：1 最大值：4294967295
enterprise_project_id	String	实例所属企业项目ID 最小长度：36 最大长度：36
device_id	String	归属的设备ID
redundant_device_id	String	归属的冗余设备ID
public_border_group	String	归属的可用区对应的边界组(public border group)，标识是否homezone局点。
tags	Array of Tag objects	标签信息 数组长度：0 - 10

表 5-71 Tag

参数	参数类型	描述
key	String	标签键，最大长度36个unicode字符，格式为大小写字母，数字，中划线“-”，下划线“_”，中文。 最小长度： 0 最大长度： 36
value	String	标签值，最大长度43个unicode字符，格式为大小写字母，数字，中划线“-”，下划线“_”，点“.”，中文。 最小长度： 0 最大长度： 43

表 5-72 PageInfo

参数	参数类型	描述
previous_marker	String	上一页的marker，值为资源的uuid 最小长度： 0 最大长度： 36
current_count	Integer	当前列表中资源数量 最小值： 0 最大值： 2000
next_marker	String	下一页的marker，值为资源的uuid，为空时表示最后一页 最小长度： 0 最大长度： 36

请求示例

查询虚拟网关列表

```
GET https://{dc_endpoint}/v3/0605768a3300d5762f82c01180692873/dcaas/virtual-gateways
```

响应示例

状态码：**200**

OK

- 成功查询接入虚拟私有云的虚拟网关列表。

```
{  
  "virtual_gateways": [ {  
    "id": "20082c1b-3c99-48d8-8e8c-116af5d7e9f0",  
    "name": "vgw-c7b22",  
    "description": ""  
  }  
]
```

```
"tenant_id" : "0605768a3300d5762f82c01180692873",
"vpc_id" : "6592c28e-95d7-4b0a-9f61-004fdf03420c",
"device_id" : "26.151.63.100",
"redundant_device_id" : "26.152.128.20",
"type" : "default",
"status" : "ACTIVE",
"admin_state_up" : true,
"bgp_asn" : 64512,
"local_ep_group" : [ "192.168.1.0/24" ],
"enterprise_project_id" : "0",
"public_border_group" : "center"
}],
"request_id" : "765f7aaf8f2edd0e719de564ef72e2de"
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.dc.v3.region.DcRegion;
import com.huaweicloud.sdk.dc.v3.*;
import com.huaweicloud.sdk.dc.v3.model.*;

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

public class ListVirtualGatewaysSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
        // security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
        // environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running
        // this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
            .withAk(ak)
            .withSk(sk);

        DcClient client = DcClient.newBuilder()
            .withCredential(auth)
            .withRegion(DcRegion.valueOf("<YOUR REGION>"))
            .build();
        ListVirtualGatewaysRequest request = new ListVirtualGatewaysRequest();
        request.withVpclid();
        request.withLimit(<limit>);
        request.withMarker("<marker>");
        request.withFields();
        request.withSortDir();
        request.withSortKey("<sort_key>");
        request.withId();
        request.withEnterpriseProjectId();
        try {
            ListVirtualGatewaysResponse response = client.listVirtualGateways(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        }
    }
}
```

```
    } catch (RequestTimeoutException e) {
        e.printStackTrace();
    } catch (ServiceResponseException e) {
        e.printStackTrace();
        System.out.println(e.getHttpStatusCode());
        System.out.println(e.getRequestId());
        System.out.println(e.getErrorCode());
        System.out.println(e.getErrorMsg());
    }
}
}
```

Python

```
# coding: utf-8

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdc.v3.region.dc_region import DcRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdc.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    # risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    # variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this
    # example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.getenv("CLOUD_SDK_AK")
    sk = os.getenv("CLOUD_SDK_SK")

    credentials = BasicCredentials(ak, sk) \

    client = DcClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DcRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListVirtualGatewaysRequest()
        request.vpc_id =
        request.limit = <limit>
        request.marker = "<marker>"
        request.fields =
        request.sort_dir =
        request.sort_key = "<sort_key>"
        request.id =
        request.enterprise_project_id =
        response = client.list_virtual_gateways(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dc "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
```

```
risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
// In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
ak := os.Getenv("CLOUD_SDK_AK")
sk := os.Getenv("CLOUD_SDK_SK")

auth := basic.NewCredentialsBuilder().
    WithAk(ak).
    WithSk(sk).
    Build()

client := dc.NewDcClient(
    dc.DcClientBuilder().
        WithRegion(region.ValueOf("<YOUR REGION>")).
        WithCredential(auth).
        Build())

request := &model.ListVirtualGatewaysRequest{
    limitRequest:= int32(<limit>)
    request.Limit = &limitRequest
    markerRequest:= "<marker>"
    request.Marker = &markerRequest
    sortKeyRequest:= "<sort_key>"
    request.SortKey = &sortKeyRequest
    response, err := client.ListVirtualGateways(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见[API Explorer](#)的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK

错误码

请参见[错误码](#)。

5.2.5 创建虚拟网关

功能介绍

创建虚拟网关

调用方法

请参见[如何调用API](#)。

URI

POST /v3/{project_id}/dcaas/virtual-gateways

表 5-73 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	租户项目ID

请求参数

表 5-74 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。获取Token，请参考《统一身份认证服务API参考》的“获取用户Token”章节。请求响应成功后在响应消息头中包含的“X-Subject-Token”的值即为Token值。 最小长度：0 最大长度：10240

表 5-75 请求 Body 参数

参数	是否必选	参数类型	描述
virtual_gateway	否	CreateVirtualGateway object	创建虚拟网关对象参数

表 5-76 CreateVirtualGateway

参数	是否必选	参数类型	描述
vpc_id	是	String	虚拟网关接入的VPC的ID
name	否	String	虚拟网关名字 最小长度：0 最大长度：64
description	否	String	虚拟网关的描述信息 最小长度：0 最大长度：128

参数	是否必选	参数类型	描述
local_ep_group	是	Array of strings	虚拟网关到访问云上服务IPv4子网列表，通常是vpc的cidrs
local_ep_group_ipv6	否	Array of strings	预留字段用于虚拟网关到访问云上服务IPv6子网列表，通常是vpc的cidrs
bgp_asn	否	Integer	虚拟网关本地的BGP自治域号(asn) 最小值：1 最大值：4294967295
enterprise_project_id	否	String	实例所属企业项目ID 最小长度：36 最大长度：36
tags	否	Array of Tag objects	标签信息 数组长度：0 - 10

表 5-77 Tag

参数	是否必选	参数类型	描述
key	是	String	标签键，最大长度36个unicode字符，格式为大小写字母，数字，中划线“-”，下划线“_”，中文。 最小长度：0 最大长度：36
value	否	String	标签值，最大长度43个unicode字符，格式为大小写字母，数字，中划线“-”，下划线“_”，点“.”，中文。 最小长度：0 最大长度：43

响应参数

状态码：201

表 5-78 响应 Body 参数

参数	参数类型	描述
virtual_gateway	VirtualGateway object	虚拟网关对象
request_id	String	操作请求ID

表 5-79 VirtualGateway

参数	参数类型	描述
id	String	虚拟网关的ID
vpc_id	String	虚拟网关接入的VPC的ID
tenant_id	String	实例所属项目ID。 最小长度：32 最大长度：32
name	String	虚拟网关的名字 最小长度：0 最大长度：64
description	String	虚拟网关的描述 最小长度：0 最大长度：128
type	String	虚拟网关类型：default 缺省值： default
local_ep_group	Array of strings	虚拟网关到访问云上服务IPv4子网列表，通常是vpc的cidrs
local_ep_group_ipv6	Array of strings	预留字段用于虚拟网关到访问云上服务IPv6子网列表，通常是vpc的cidrs
admin_state_up	Boolean	管理状态：true或false 缺省值： true
status	String	操作状态，合法值是：ACTIVE, DOWN, BUILD, ERROR, PENDING_CREATE, PENDING_UPDATE, PENDING_DELETE
bgp_asn	Integer	虚拟网关本地的BGP自治域号(asn) 最小值：1 最大值： 4294967295

参数	参数类型	描述
enterprise_project_id	String	实例所属企业项目ID 最小长度：36 最大长度：36
device_id	String	归属的设备ID
redundant_device_id	String	归属的冗余设备ID
public_border_group	String	归属的可用区对应的边界组(public border group)，标识是否homezone局点。
tags	Array of Tag objects	标签信息 数组长度：0 - 10

表 5-80 Tag

参数	参数类型	描述
key	String	标签键，最大长度36个unicode字符，格式为大小写字母，数字，中划线“-”，下划线“_”，中文。 最小长度：0 最大长度：36
value	String	标签值，最大长度43个unicode字符，格式为大小写字母，数字，中划线“-”，下划线“_”，点“.”，中文。 最小长度：0 最大长度：43

请求示例

- 创建一个接入虚拟私有云的虚拟网关，设置本地BGP ASN为64512，虚拟网关访问云专服务IPv4子网列表为192.168.1.0/24。

```
POST https://{dc_endpoint}/v3/0605768a3300d5762f82c01180692873/dcaas/virtual-gateways
```

```
{
  "virtual_gateway": {
    "name": "vgw-c7b22",
    "description": "",
    "vpc_id": "6592c28e-95d7-4b0a-9f61-004fdf03420c",
    "bgp_asn": 64512,
    "local_ep_group": [ "192.168.1.0/24" ]
  }
}
```

响应示例

状态码： 201

Created

- 成功创建接入虚拟私有云的虚拟网关。

```
{
  "virtual_gateway": {
    "id": "20082c1b-3c99-48d8-8e8c-116af5d7e9f0",
    "name": "vgw-c7b22",
    "description": "",
    "tenant_id": "0605768a3300d5762f82c01180692873",
    "vpc_id": "6592c28e-95d7-4b0a-9f61-004fdf03420c",
    "device_id": "26.151.63.100",
    "redundant_device_id": "26.152.128.20",
    "type": "default",
    "status": "ACTIVE",
    "admin_state_up": true,
    "bgp_asn": 64512,
    "local_ep_group": [ "192.168.1.0/24" ],
    "enterprise_project_id": "0",
    "public_border_group": "center"
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

- 创建一个接入虚拟私有云的虚拟网关，设置本地BGP ASN为64512，虚拟网关访问云专服务IPv4子网列表为192.168.1.0/24。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.dc.v3.region.DcRegion;
import com.huaweicloud.sdk.dc.v3.*;
import com.huaweicloud.sdk.dc.v3.model.*;

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

public class CreateVirtualGatewaySolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
        // security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
        // environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before
        // running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local
        // environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
            .withAk(ak)
            .withSk(sk);

        DcClient client = DcClient.newBuilder()
            .withCredential(auth)
            .withRegion(DcRegion.valueOf("<YOUR REGION>"))
            .build();
        CreateVirtualGatewayRequest request = new CreateVirtualGatewayRequest();
        CreateVirtualGatewayRequestBody body = new CreateVirtualGatewayRequestBody();
        List<String> listVirtualGatewayLocalEpGroup = new ArrayList<>();
```

```
listVirtualGatewayLocalEpGroup.add("192.168.1.0/24");
CreateVirtualGateway virtualGatewaybody = new CreateVirtualGateway();
virtualGatewaybody.withVpclid("6592c28e-95d7-4b0a-9f61-004fdf03420c")
    .withName("vgw-c7b22")
    .withDescription("")
    .withLocalEpGroup(listVirtualGatewayLocalEpGroup)
    .withBgpAsn(64512);
body.withVirtualGateway(virtualGatewaybody);
request.withBody(body);
try {
    CreateVirtualGatewayResponse response = client.createVirtualGateway(request);
    System.out.println(response.toString());
} catch (ConnectionException e) {
    e.printStackTrace();
} catch (RequestTimeoutException e) {
    e.printStackTrace();
} catch (ServiceResponseException e) {
    e.printStackTrace();
    System.out.println(e.getHttpStatusCode());
    System.out.println(e.getRequestId());
    System.out.println(e.getErrorCode());
    System.out.println(e.getErrorMsg());
}
}
```

- 创建一个接入企业路由器的虚拟网关，设置BGP ASN为64512。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.dc.v3.region.DcRegion;
import com.huaweicloud.sdk.dc.v3.*;
import com.huaweicloud.sdk.dc.v3.model.*;

public class CreateVirtualGatewaySolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
        // security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
        // environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before
        // running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local
        // environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
            .withAk(ak)
            .withSk(sk);

        DcClient client = DcClient.newBuilder()
            .withCredential(auth)
            .withRegion(DcRegion.valueOf("<YOUR REGION>"))
            .build();
        CreateVirtualGatewayRequest request = new CreateVirtualGatewayRequest();
        CreateVirtualGatewayRequestBody body = new CreateVirtualGatewayRequestBody();
        CreateVirtualGateway virtualGatewaybody = new CreateVirtualGateway();
        virtualGatewaybody.withName("vgw-er")
            .withDescription("")
            .withBgpAsn(64512);
        body.withVirtualGateway(virtualGatewaybody);
        request.withBody(body);
        try {
            CreateVirtualGatewayResponse response = client.createVirtualGateway(request);
            System.out.println(response.toString());
        }
    }
}
```

```
    } catch (ConnectionException e) {
        e.printStackTrace();
    } catch (RequestTimeoutException e) {
        e.printStackTrace();
    } catch (ServiceResponseException e) {
        e.printStackTrace();
        System.out.println(e.getHttpStatusCode());
        System.out.println(e.getRequestId());
        System.out.println(e.getErrorCode());
        System.out.println(e.getErrorMsg());
    }
}
}
```

Python

- 创建一个接入虚拟私有云的虚拟网关，设置本地BGP ASN为64512，虚拟网关访问云专服务IPv4子网列表为192.168.1.0/24。

```
# coding: utf-8
```

```
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdc.v3.region.dc_region import DcRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdc.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
    # security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
    # environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before
    # running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local
    # environment
    ak = os.getenv("CLOUD_SDK_AK")
    sk = os.getenv("CLOUD_SDK_SK")

    credentials = BasicCredentials(ak, sk) \

    client = DcClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DcRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = CreateVirtualGatewayRequest()
        listLocalEpgroupVirtualGateway = [
            "192.168.1.0/24"
        ]
        virtualGatewaybody = CreateVirtualGateway(
            vpc_id="6592c28e-95d7-4b0a-9f61-004fdf03420c",
            name="vgw-c7b22",
            description="",
            local_ep_group=listLocalEpgroupVirtualGateway,
            bgp_asn=64512
        )
        request.body = CreateVirtualGatewayRequestBody(
            virtual_gateway=virtualGatewaybody
        )
        response = client.create_virtual_gateway(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

- 创建一个接入企业路由器的虚拟网关，设置BGP ASN为64512。

```
# coding: utf-8
```

```
from huaweicloudsdkcore.auth.credentials import BasicCredentials
```

```
from huaweicloudsdkdc.v3.region.dc_region import DcRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdc.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
    # security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
    # environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before
    # running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local
    # environment
    ak = os.getenv("CLOUD_SDK_AK")
    sk = os.getenv("CLOUD_SDK_SK")

    credentials = BasicCredentials(ak, sk) \

    client = DcClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DcRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = CreateVirtualGatewayRequest()
        virtualGatewaybody = CreateVirtualGateway(
            name="vgw-er",
            description="",
            bgp_asn=64512
        )
        request.body = CreateVirtualGatewayRequestBody(
            virtual_gateway=virtualGatewaybody
        )
        response = client.create_virtual_gateway(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

- 创建一个接入虚拟私有云的虚拟网关，设置本地BGP ASN为64512，虚拟网关访问云专服务IPv4子网列表为192.168.1.0/24。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dc "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
    // security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
    // environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before
    // running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local
    // environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := dc.NewDcClient(
```



```
dc.DcClientBuilder().
    WithRegion(region.ValueOf("<YOUR REGION>")).
    WithCredential(auth).
    Build()

request := &model.CreateVirtualGatewayRequest{}
var listLocalEpGroupVirtualGateway = []string{
    "192.168.1.0/24",
}
nameVirtualGateway:= "vgw-c7b22"
descriptionVirtualGateway:= ""
bgpAsnVirtualGateway:= int32(64512)
virtualGatewaybody := &model.CreateVirtualGateway{
    Vpclid: "6592c28e-95d7-4b0a-9f61-004fdf03420c",
    Name: &nameVirtualGateway,
    Description: &descriptionVirtualGateway,
    LocalEpGroup: listLocalEpGroupVirtualGateway,
    BgpAsn: &bgpAsnVirtualGateway,
}
request.Body = &model.CreateVirtualGatewayRequestBody{
    VirtualGateway: virtualGatewaybody,
}
response, err := client.CreateVirtualGateway(request)
if err == nil {
    fmt.Printf("%+v\n", response)
} else {
    fmt.Println(err)
}
}
```

- 创建一个接入企业路由器的虚拟网关，设置BGP ASN为64512。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dc "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
    // security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
    // environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before
    // running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local
    // environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := dc.NewDcClient(
        dc.DcClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build()

    request := &model.CreateVirtualGatewayRequest{}
    nameVirtualGateway:= "vgw-er"
    descriptionVirtualGateway:= ""
    bgpAsnVirtualGateway:= int32(64512)
    virtualGatewaybody := &model.CreateVirtualGateway{
        Name: &nameVirtualGateway,
        Description: &descriptionVirtualGateway,
        BgpAsn: &bgpAsnVirtualGateway,
```

```
}
request.Body = &model.CreateVirtualGatewayRequestBody{
    VirtualGateway: virtualGatewaybody,
}
response, err := client.CreateVirtualGateway(request)
if err == nil {
    fmt.Printf("%+v\n", response)
} else {
    fmt.Println(err)
}
}
```

更多

更多编程语言的SDK代码示例，请参见[API Explorer](#)的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
201	Created

错误码

请参见[错误码](#)。

5.3 虚拟接口

5.3.1 查询虚拟接口详情

功能介绍

查询虚拟接口详细信息

调用方法

请参见[如何调用API](#)。

URI

GET /v3/{project_id}/dcaas/virtual-interfaces/{virtual_interface_id}

表 5-81 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	租户项目ID

参数	是否必选	参数类型	描述
virtual_interface_id	是	String	虚拟接口ID。 最小长度：36 最大长度：36

表 5-82 Query 参数

参数	是否必选	参数类型	描述
fields	否	Array	显示字段列表 数组长度：1 - 5

请求参数

表 5-83 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。获取Token，请参考《统一身份认证服务API参考》的“获取用户Token”章节。请求响应成功后在响应消息头中包含的“X-Subject-Token”的值即为Token值。 最小长度：0 最大长度：10240

响应参数

状态码：200

表 5-84 响应 Body 参数

参数	参数类型	描述
request_id	String	操作请求ID
virtual_interface	VirtualInterface object	虚拟接口对象

表 5-85 VirtualInterface

参数	参数类型	描述
id	String	虚拟接口的ID 最大长度：36
name	String	虚拟接口的名字 最大长度：64
admin_state_up	Boolean	管理状态：true或false
bandwidth	Integer	虚拟接口接入带宽 最小值：2 最大值：2147483647
create_time	String	虚拟接口创建时间。采用UTC时间格式，格式为：yyyy-MM-ddTHH:mm:ss.SSSZ 最大长度：255
description	String	虚拟接口的描述 最大长度：128
direct_connect_id	String	物理专线的ID 最大长度：36
service_type	String	接入网关的类型：包括VGW、GDGW、LGW等枚举值： <ul style="list-style-type: none">• VGW• GDGW• LGW
status	String	操作状态，合法值是：ACTIVE, DOWN, BUILD, ERROR, PENDING_CREATE, PENDING_UPDATE, PENDING_DELETE, DELETED, AUTHORIZATION, REJECTED
tenant_id	String	租户ID 最小长度：32 最大长度：32
type	String	表示接口类型：private 缺省值：private 最大长度：255 枚举值： <ul style="list-style-type: none">• private• public

参数	参数类型	描述
vgw_id	String	虚拟网关的ID 最小长度：36 最大长度：36
vlan	Integer	同用户网关对接的vlan, 配置范围0-3999 最小值：0 最大值：3999
route_limit	Integer	VIF远端子网路由配置规格 最小值：1 最大值：200 缺省值：50
enable_nqa	Boolean	是否使能nqa功能：true或false
enable_bfd	Boolean	是否使能bfd功能：true或false
lag_id	String	VIF关联的链路聚合组ID 最小长度：36 最大长度：36
device_id	String	归属的设备ID
enterprise_project_id	String	实例所属企业项目ID 最小长度：36 最大长度：36
tags	Array of Tag objects	标签信息 数组长度：0 - 10
local_gateway_v4_ip	String	云侧网关IPv4接口地址，该字段现已经移到vifpeer参数列表中，未来将会废弃。
remote_gateway_v4_ip	String	客户侧网关IPv4接口地址，该字段现已经移到vifpeer参数列表中，未来将会废弃。
ies_id	String	归属的IES站点的ID
reason	String	如果资源的状态是Error的情况下，该参数会显示相关错误信息。
rate_limit	Boolean	标识虚拟接口是否开启限速
address_family	String	接口的地址簇类型，ipv4，ipv6。该字段现已迁移到vifpeer参数列表中，未来将会废弃。
local_gateway_v6_ip	String	云侧网关IPv6接口地址，该字段现已迁移到vifpeer参数列表中，未来将会废弃。
remote_gateway_v6_ip	String	客户侧网关IPv6接口地址，该字段现已迁移到vifpeer参数列表中，未来将会废弃。

参数	参数类型	描述
lgw_id	String	本地网关的ID，用于IES场景。
gateway_id	String	虚拟接口关联的网关的ID
remote_ep_group	Array of strings	远端子网列表，记录租户侧的cidrs。该字段现已迁移到vifpeer参数列表中，未来将会废弃。
service_ep_group	Array of strings	该字段用于公网专线接口，表示租户可以访问云上公网服务地址列表。该字段现已迁移到vifpeer参数列表中，未来将会废弃。
bgp_route_limit	Integer	BGP的路由配置规格
priority	String	虚拟接口的优先级，支持两种优先级状态normal和low。接口优先级相同时表示负载关系，接口优先级不同时表示主备关系，出云流量优先转到优先级更高的normal接口。目前仅BGP模式接口支持。 缺省值： normal 枚举值： <ul style="list-style-type: none"> • normal • low
vif_peers	Array of VifPeer objects	vif的Peer的相关信息
extend_attribute	VifExtendAttribute object	扩展参数信息

表 5-86 Tag

参数	参数类型	描述
key	String	标签键，最大长度36个unicode字符，格式为大小写字母，数字，中划线“-”，下划线“_”，中文。 最小长度： 0 最大长度： 36
value	String	标签值，最大长度43个unicode字符，格式为大小写字母，数字，中划线“-”，下划线“_”，点“.”，中文。 最小长度： 0 最大长度： 43

表 5-87 VifPeer

参数	参数类型	描述
id	String	资源ID 最小长度：36 最大长度：36
tenant_id	String	归属租户ID 最小长度：36 最大长度：36
name	String	VIF对等体名字 最小长度：0 最大长度：64
description	String	VIF对等体名字描述信息 最小长度：0 最大长度：128
address_family	String	接口的地址簇类型， ipv4, ipv6
local_gateway_ip	String	VIF对等体云侧接口地址
remote_gateway_ip	String	VIF对等体客户侧接口地址
route_mode	String	路由模式： static/bgp 最大长度：255 枚举值： <ul style="list-style-type: none">• bgp• static
bgp_asn	Integer	BGP邻居的AS号 最小值：1 最大值：4294967295
bgp_md5	String	BGP邻居的MD5密码
remote_ep_group	Array of strings	远端子网列表，记录租户侧的cidrs
service_ep_group	Array of strings	该字段用于公网专线接口,表示租户可以访问云上公网服务地址列表
device_id	String	归属的设备ID
bgp_route_limit	Integer	BGP的路由配置规格

参数	参数类型	描述
bgp_status	String	接口BGP协议状态,如果是静态路由接口则状态为null 最大长度: 10
status	String	VIF对等体状态
vif_id	String	vif对等体对应的虚拟接口ID 最小长度: 36 最大长度: 36
receive_route_num	Integer	路由模式为bgp: receive_route_num值为接收搭配BGP的路由数目; 路由模式为static: 该字段无意义, 值为-1; 注: 若早期接入云上的部分用户无法获取该字段信息, 如需要请联系客服迁移专线端口。
enable_nqa	Boolean	是否使能nqa功能: true或false
enable_bfd	Boolean	是否使能bfd功能: true或false

表 5-88 VifExtendAttribute

参数	参数类型	描述
ha_type	String	虚拟接口的可用性检测类型 枚举值: <ul style="list-style-type: none">• nqa• bfd
ha_mode	String	检测的具体的配置模式 枚举值: <ul style="list-style-type: none">• auto_single• auto_multi• static_single• static_multi• enhance_nqa
detect_multiplier	Integer	检测的重试次数 缺省值: 5
min_rx_interval	Integer	检测的接收时长间隔 缺省值: 1000
min_tx_interval	Integer	检测的发送时长间隔 缺省值: 1000

参数	参数类型	描述
remote_disclaim	Integer	检测的远端的标识，用于静态BFD
local_disclaim	Integer	检测的本端的标识，用于静态BFD

请求示例

查询虚拟接口

```
POST https://{dc_endpoint}/v3/0605768a3300d5762f82c01180692873/dcaas/virtual-interfaces/0d0fdf63-f2c4-491c-8866-d504796189be
```

响应示例

状态码： 200

OK

```
{
  "virtual_interface": {
    "id": "0d0fdf63-f2c4-491c-8866-d504796189be",
    "name": "vif-0819",
    "description": "",
    "tenant_id": "0605768a3300d5762f82c01180692873",
    "direct_connect_id": "4673e339-8412-4ee1-b73e-2ba9cdfa54c1",
    "vgw_id": "8a47064a-f34c-4f94-b7fe-cac456c9b37b",
    "type": "private",
    "service_type": "VGW",
    "vlan": 332,
    "bandwidth": 2,
    "status": "ACTIVE",
    "create_time": "2022-08-19T11:28:06.000Z",
    "admin_state_up": true,
    "enable_bfd": false,
    "route_limit": 50,
    "enable_nqa": false,
    "local_gateway_v4_ip": "1.1.1.1/30",
    "remote_gateway_v4_ip": "1.1.1.2/30",
    "ies_id": null,
    "reason": null,
    "rate_limit": false,
    "address_family": "ipv4",
    "local_gateway_v6_ip": null,
    "remote_gateway_v6_ip": null,
    "lgw_id": null,
    "lag_id": null,
    "gateway_id": null,
    "remote_ep_group": [ "1.1.2.0/30" ],
    "service_ep_group": [ ],
    "bgp_route_limit": 100,
    "priority": "normal",
    "vif_peers": [ {
      "id": "c768eb52-12a8-4859-9b43-81194643040c",
      "tenant_id": "0605768a3300d5762f82c01180692873",
      "name": "vif-0819",
      "description": "",
      "address_family": "ipv4",
      "local_gateway_ip": "1.1.1.1/30",
      "remote_gateway_ip": "1.1.1.2/30",
      "route_mode": "static",
      "bgp_asn": null,
    }
  ]
}
```

```
"bgp_md5" : null,
"device_id" : "18.9.215.131",
"bgp_route_limit" : 100,
"bgp_status" : null,
"status" : "ACTIVE",
"vif_id" : "0d0fdf63-f2c4-491c-8866-d504796189be",
"receive_route_num" : -1,
"remote_ep_group" : [ "1.1.2.0/30" ],
"service_ep_group" : null,
"enable_bfd" : false,
"enable_nqa" : false
}],
"enterprise_project_id" : "0"
},
"request_id" : "5633df7af874576d819a481c76673236"
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.dc.v3.region.DcRegion;
import com.huaweicloud.sdk.dc.v3.*;
import com.huaweicloud.sdk.dc.v3.model.*;

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

public class ShowVirtualInterfaceSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
        // security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
        // environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running
        // this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
            .withAk(ak)
            .withSk(sk);

        DcClient client = DcClient.newBuilder()
            .withCredential(auth)
            .withRegion(DcRegion.valueOf("<YOUR REGION>"))
            .build();
        ShowVirtualInterfaceRequest request = new ShowVirtualInterfaceRequest();
        request.withFields();
        try {
            ShowVirtualInterfaceResponse response = client.showVirtualInterface(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
        }
    }
}
```

```
        System.out.println(e.getRequestId());
        System.out.println(e.getErrorCode());
        System.out.println(e.getErrorMsg());
    }
}
}
```

Python

```
# coding: utf-8

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdc.v3.region.dc_region import DcRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdc.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    # risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    # variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this
    # example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.getenv("CLOUD_SDK_AK")
    sk = os.getenv("CLOUD_SDK_SK")

    credentials = BasicCredentials(ak, sk) \

    client = DcClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DcRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowVirtualInterfaceRequest()
        request.fields =
        response = client.show_virtual_interface(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dc "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    // risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    // variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this
    // example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()
```

```
client := dc.NewDcClient(  
    dc.DcClientBuilder().  
        WithRegion(region.ValueOf("<YOUR REGION>")).  
        WithCredential(auth).  
        Build())  
  
request := &model.ShowVirtualInterfaceRequest{}  
response, err := client.ShowVirtualInterface(request)  
if err == nil {  
    fmt.Printf("%+v\n", response)  
} else {  
    fmt.Println(err)  
}  
}
```

更多

更多编程语言的SDK代码示例，请参见[API Explorer](#)的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK

错误码

请参见[错误码](#)。

5.3.2 更新虚拟接口

功能介绍

更新虚拟接口的详细信息

调用方法

请参见[如何调用API](#)。

URI

PUT /v3/{project_id}/dcaas/virtual-interfaces/{virtual_interface_id}

表 5-89 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	租户项目ID
virtual_interface_id	是	String	虚拟接口ID。 最小长度：36 最大长度：36

请求参数

表 5-90 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。获取Token，请参考《统一身份认证服务API参考》的“获取用户Token”章节。请求响应成功后在响应消息头中包含的“X-Subject-Token”的值即为Token值。 最小长度：0 最大长度：10240

表 5-91 请求 Body 参数

参数	是否必选	参数类型	描述
virtual_interface	是	UpdateVirtualInterface object	虚拟接口更新对象

表 5-92 UpdateVirtualInterface

参数	是否必选	参数类型	描述
name	否	String	虚拟接口名字 最小长度：0 最大长度：64
description	否	String	虚拟接口描述信息 最小长度：0 最大长度：128
bandwidth	否	Integer	虚拟接口带宽配置 最小值：2 最大值：2147483647
remote_ep_group	否	Array of strings	远端子网列表，记录租户侧的cidrs
service_ep_group	否	Array of strings	用于公网专线,用户访问公网服务地址列表。
enable_bfd	否	Boolean	是否使能bfd功能：true或false。

参数	是否必选	参数类型	描述
enable_nqa	否	Boolean	是否使能nqa功能: true或false。
status	否	String	对其他租户创建的虚拟接口进行确认,可以是ACCEPTED和REJECTED 枚举值: <ul style="list-style-type: none">● ACCEPTED● REJECTED

响应参数

状态码: 200

表 5-93 响应 Body 参数

参数	参数类型	描述
request_id	String	操作请求ID
virtual_interface	VirtualInterface object	虚拟接口对象

表 5-94 VirtualInterface

参数	参数类型	描述
id	String	虚拟接口的ID 最大长度: 36
name	String	虚拟接口的名字 最大长度: 64
admin_state_up	Boolean	管理状态: true或false
bandwidth	Integer	虚拟接口接入带宽 最小值: 2 最大值: 2147483647
create_time	String	虚拟接口创建时间。采用UTC时间格式, 格式为: yyyy-MM-ddTHH:mm:ss.SSSZ 最大长度: 255
description	String	虚拟接口的描述 最大长度: 128

参数	参数类型	描述
direct_connect_id	String	物理专线的ID 最大长度：36
service_type	String	接入网关的类型：包括VGW、GDGW、LGW等 枚举值： <ul style="list-style-type: none">• VGW• GDGW• LGW
status	String	操作状态，合法值是：ACTIVE, DOWN, BUILD, ERROR, PENDING_CREATE, PENDING_UPDATE, PENDING_DELETE, DELETED, AUTHORIZATION, REJECTED
tenant_id	String	租户ID 最小长度：32 最大长度：32
type	String	表示接口类型：private 缺省值： private 最大长度：255 枚举值： <ul style="list-style-type: none">• private• public
vgw_id	String	虚拟网关的ID 最小长度：36 最大长度：36
vlan	Integer	同用户网关对接的vlan, 配置范围0-3999 最小值：0 最大值：3999
route_limit	Integer	VIF远端子网路由配置规格 最小值：1 最大值：200 缺省值：50
enable_nqa	Boolean	是否使能nqa功能：true或false
enable_bfd	Boolean	是否使能bfd功能：true或false
lag_id	String	VIF关联的链路聚合组ID 最小长度：36 最大长度：36
device_id	String	归属的设备ID

参数	参数类型	描述
enterprise_project_id	String	实例所属企业项目ID 最小长度：36 最大长度：36
tags	Array of Tag objects	标签信息 数组长度：0 - 10
local_gateway_v4_ip	String	云侧网关IPv4接口地址，该字段现已经移到vifpeer参数列表中，未来将会废弃。
remote_gateway_v4_ip	String	客户侧网关IPv4接口地址，该字段现已经移到vifpeer参数列表中，未来将会废弃。
ies_id	String	归属的IES站点的ID
reason	String	如果资源的状态是Error的情况下，该参数会显示相关错误信息。
rate_limit	Boolean	标识虚拟接口是否开启限速
address_family	String	接口的地址簇类型，ipv4，ipv6。该字段现已迁移到vifpeer参数列表中，未来将会废弃。
local_gateway_v6_ip	String	云侧网关IPv6接口地址，该字段现已迁移到vifpeer参数列表中，未来将会废弃。
remote_gateway_v6_ip	String	客户侧网关IPv6接口地址，该字段现已迁移到vifpeer参数列表中，未来将会废弃。
lgw_id	String	本地网关的ID，用于IES场景。
gateway_id	String	虚拟接口关联的网关的ID
remote_ep_group	Array of strings	远端子网列表，记录租户侧的cidrs。该字段现已迁移到vifpeer参数列表中，未来将会废弃。
service_ep_group	Array of strings	该字段用于公网专线接口，表示租户可以访问云上公网服务地址列表。该字段现已迁移到vifpeer参数列表中，未来将会废弃。
bgp_route_limit	Integer	BGP的路由配置规格
priority	String	虚拟接口的优先级，支持两种优先级状态normal和low。接口优先级相同时表示负载关系，接口优先级不同时表示主备关系，出云流量优先转到优先级更高的normal接口。目前仅BGP模式接口支持。 缺省值： normal 枚举值： <ul style="list-style-type: none">● normal● low

参数	参数类型	描述
vif_peers	Array of VifPeer objects	vif的Peer的相关信息
extend_attribute	VifExtendAttribute object	扩展参数信息

表 5-95 Tag

参数	参数类型	描述
key	String	标签键，最大长度36个unicode字符，格式为大小写字母，数字，中划线“-”，下划线“_”，中文。 最小长度：0 最大长度：36
value	String	标签值，最大长度43个unicode字符，格式为大小写字母，数字，中划线“-”，下划线“_”，点“.”，中文。 最小长度：0 最大长度：43

表 5-96 VifPeer

参数	参数类型	描述
id	String	资源ID 最小长度：36 最大长度：36
tenant_id	String	归属租户ID 最小长度：36 最大长度：36
name	String	VIF对等体名字 最小长度：0 最大长度：64
description	String	VIF对等体名字描述信息 最小长度：0 最大长度：128
address_family	String	接口的地址簇类型，ipv4，ipv6

参数	参数类型	描述
local_gateway_ip	String	VIF对等体云侧接口地址
remote_gateway_ip	String	VIF对等体客户侧接口地址
route_mode	String	路由模式：static/bgp 最大长度：255 枚举值： <ul style="list-style-type: none">• bgp• static
bgp_asn	Integer	BGP邻居的AS号 最小值：1 最大值：4294967295
bgp_md5	String	BGP邻居的MD5密码
remote_ep_group	Array of strings	远端子网列表，记录租户侧的cidrs
service_ep_group	Array of strings	该字段用于公网专线接口,表示租户可以访问云上公网服务地址列表
device_id	String	归属的设备ID
bgp_route_limit	Integer	BGP的路由配置规格
bgp_status	String	接口BGP协议状态,如果是静态路由接口则状态为null 最大长度：10
status	String	VIF对等体状态
vif_id	String	vif对等体对应的虚拟接口ID 最小长度：36 最大长度：36
receive_route_num	Integer	路由模式为bgp：receive_route_num值为接收搭配BGP的路由数目；路由模式为static：该字段无意义，值为-1；注：若早期接入云上的部分用户无法获取该字段信息，如需要请联系客服迁移专线端口。
enable_nqa	Boolean	是否使能nqa功能：true或false
enable_bfd	Boolean	是否使能bfd功能：true或false

表 5-97 VifExtendAttribute

参数	参数类型	描述
ha_type	String	虚拟接口的可用性检测类型 枚举值： <ul style="list-style-type: none">• nqa• bfd
ha_mode	String	检测的具体的配置模式 枚举值： <ul style="list-style-type: none">• auto_single• auto_multi• static_single• static_multi• enhance_nqa
detect_multiplier	Integer	检测的重试次数 缺省值：5
min_rx_interval	Integer	检测的接收时长间隔 缺省值：1000
min_tx_interval	Integer	检测的发送时长间隔 缺省值：1000
remote_disclaim	Integer	检测的远端的标识，用于静态BFD
local_disclaim	Integer	检测的本端的标识，用于静态BFD

请求示例

更新虚拟接口的名称和描述，将带宽修改为2。

```
PUT https://{dc_endpoint}/v3/0605768a3300d5762f82c01180692873/dcaas/virtual-interfaces/0d0fdf63-f2c4-491c-8866-d504796189be
```

```
{  
  "virtual_interface": {  
    "name": "vif-0819",  
    "description": "mytest",  
    "bandwidth": 2  
  }  
}
```

响应示例

状态码： 200

OK

```
{  
  "virtual_interface": {
```

```
{
  "id": "0d0fdf63-f2c4-491c-8866-d504796189be",
  "name": "vif-0819",
  "description": "mytest",
  "tenant_id": "0605768a3300d5762f82c01180692873",
  "direct_connect_id": "4673e339-8412-4ee1-b73e-2ba9cdfa54c1",
  "vgw_id": "8a47064a-f34c-4f94-b7fe-cac456c9b37b",
  "type": "private",
  "service_type": "VGW",
  "vlan": 332,
  "bandwidth": 2,
  "status": "ACTIVE",
  "create_time": "2022-08-19T11:28:06.000Z",
  "admin_state_up": true,
  "enable_bfd": false,
  "route_limit": 50,
  "enable_nqa": false,
  "local_gateway_v4_ip": "1.1.1.1/30",
  "remote_gateway_v4_ip": "1.1.1.2/30",
  "ies_id": null,
  "reason": null,
  "rate_limit": false,
  "address_family": "ipv4",
  "local_gateway_v6_ip": null,
  "remote_gateway_v6_ip": null,
  "lgw_id": null,
  "lag_id": null,
  "gateway_id": null,
  "remote_ep_group": [ "1.1.2.0/30" ],
  "service_ep_group": [ ],
  "bgp_route_limit": 100,
  "priority": "normal",
  "vif_peers": [ {
    "id": "c768eb52-12a8-4859-9b43-81194643040c",
    "tenant_id": "0605768a3300d5762f82c01180692873",
    "name": "vif-0819",
    "description": "",
    "address_family": "ipv4",
    "local_gateway_ip": "1.1.1.1/30",
    "remote_gateway_ip": "1.1.1.2/30",
    "route_mode": "static",
    "bgp_asn": null,
    "bgp_md5": null,
    "device_id": "18.9.215.131",
    "bgp_route_limit": 100,
    "bgp_status": null,
    "status": "ACTIVE",
    "vif_id": "0d0fdf63-f2c4-491c-8866-d504796189be",
    "receive_route_num": -1,
    "remote_ep_group": [ "1.1.2.0/30" ],
    "service_ep_group": null,
    "enable_bfd": false,
    "enable_nqa": false
  } ],
  "enterprise_project_id": "0"
},
"request_id": "5633df7af874576d819a481c76673236"
}
```

SDK 代码示例

SDK代码示例如下。

Java

更新虚拟接口的名称和描述，将带宽修改为2。

```
package com.huaweicloud.sdk.test;
```

```
import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.dc.v3.region.DcRegion;
import com.huaweicloud.sdk.dc.v3.*;
import com.huaweicloud.sdk.dc.v3.model.*;

public class UpdateVirtualInterfaceSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
        // security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
        // environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running
        // this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
            .withAk(ak)
            .withSk(sk);

        DcClient client = DcClient.newBuilder()
            .withCredential(auth)
            .withRegion(DcRegion.valueOf("<YOUR REGION>"))
            .build();
        UpdateVirtualInterfaceRequest request = new UpdateVirtualInterfaceRequest();
        UpdateVirtualInterfaceRequestBody body = new UpdateVirtualInterfaceRequestBody();
        UpdateVirtualInterface virtualInterfacebody = new UpdateVirtualInterface();
        virtualInterfacebody.setName("vif-0819")
            .withDescription("mytest")
            .withBandwidth(2);
        body.withVirtualInterface(virtualInterfacebody);
        request.withBody(body);
        try {
            UpdateVirtualInterfaceResponse response = client.updateVirtualInterface(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

更新虚拟接口的名称和描述，将带宽修改为2。

```
# coding: utf-8

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdc.v3.region.dc_region import DcRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdc.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    # risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    # variables and decrypted during use to ensure security.
```

```
# In this example, AK and SK are stored in environment variables for authentication. Before running this
example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
ak = os.getenv("CLOUD_SDK_AK")
sk = os.getenv("CLOUD_SDK_SK")

credentials = BasicCredentials(ak, sk) \

client = DcClient.new_builder() \
    .with_credentials(credentials) \
    .with_region(DcRegion.value_of("<YOUR REGION>")) \
    .build()

try:
    request = UpdateVirtualInterfaceRequest()
    virtualInterfacebody = UpdateVirtualInterface(
        name="vif-0819",
        description="mytest",
        bandwidth=2
    )
    request.body = UpdateVirtualInterfaceRequestBody(
        virtual_interface=virtualInterfacebody
    )
    response = client.update_virtual_interface(request)
    print(response)
except exceptions.ClientRequestException as e:
    print(e.status_code)
    print(e.request_id)
    print(e.error_code)
    print(e.error_msg)
```

Go

更新虚拟接口的名称和描述，将带宽修改为2。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dc "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this
    example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := dc.NewDcClient(
        dc.DcClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.UpdateVirtualInterfaceRequest{}
    nameVirtualInterface:= "vif-0819"
    descriptionVirtualInterface:= "mytest"
    bandwidthVirtualInterface:= int32(2)
    virtualInterfacebody := &model.UpdateVirtualInterface{
```

```
Name: &nameVirtualInterface,  
Description: &descriptionVirtualInterface,  
Bandwidth: &bandwidthVirtualInterface,  
}  
request.Body = &model.UpdateVirtualInterfaceRequestBody{  
    VirtualInterface: virtualInterfacebody,  
}  
response, err := client.UpdateVirtualInterface(request)  
if err == nil {  
    fmt.Printf("%+v\n", response)  
} else {  
    fmt.Println(err)  
}  
}
```

更多

更多编程语言的SDK代码示例，请参见[API Explorer](#)的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK

错误码

请参见[错误码](#)。

5.3.3 删除虚拟接口

功能介绍

删除虚拟接口

调用方法

请参见[如何调用API](#)。

URI

DELETE /v3/{project_id}/dcaas/virtual-interfaces/{virtual_interface_id}

表 5-98 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	租户项目ID
virtual_interface_id	是	String	虚拟接口ID。 最小长度：36 最大长度：36

请求参数

表 5-99 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。获取Token，请参考《统一身份认证服务API参考》的“获取用户Token”章节。请求响应成功后在响应消息头中包含的“X-Subject-Token”的值即为Token值。 最小长度： 0 最大长度： 10240

响应参数

无

请求示例

删除虚拟接口

```
DELETE https://{dc_endpoint}/v3/0605768a3300d5762f82c01180692873/dcaas/virtual-interfaces/0d0fdf63-f2c4-491c-8866-d504796189be
```

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.dc.v3.region.DcRegion;
import com.huaweicloud.sdk.dc.v3.*;
import com.huaweicloud.sdk.dc.v3.model.*;

public class DeleteVirtualInterfaceSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
        // security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
        // environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running
        // this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
```



```
String sk = System.getenv("CLOUD_SDK_SK");

ICredential auth = new BasicCredentials()
    .withAk(ak)
    .withSk(sk);

DcClient client = DcClient.newBuilder()
    .withCredential(auth)
    .withRegion(DcRegion.valueOf("<YOUR REGION>"))
    .build();
DeleteVirtualInterfaceRequest request = new DeleteVirtualInterfaceRequest();
try {
    DeleteVirtualInterfaceResponse response = client.deleteVirtualInterface(request);
    System.out.println(response.toString());
} catch (ConnectionException e) {
    e.printStackTrace();
} catch (RequestTimeoutException e) {
    e.printStackTrace();
} catch (ServiceResponseException e) {
    e.printStackTrace();
    System.out.println(e.getHttpStatusCode());
    System.out.println(e.getRequestId());
    System.out.println(e.getErrorCode());
    System.out.println(e.getErrorMsg());
}
}
```

Python

```
# coding: utf-8

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdc.v3.region.dc_region import DcRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdc.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    # risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    # variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this
    # example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.getenv("CLOUD_SDK_AK")
    sk = os.getenv("CLOUD_SDK_SK")

    credentials = BasicCredentials(ak, sk) \

    client = DcClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DcRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = DeleteVirtualInterfaceRequest()
        response = client.delete_virtual_interface(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
```

```
"github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
dc "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3"
"github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/model"
region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this
    example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := dc.NewDcClient(
        dc.DcClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.DeleteVirtualInterfaceRequest{}
    response, err := client.DeleteVirtualInterface(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见[API Explorer](#)的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
204	No Content

错误码

请参见[错误码](#)。

5.3.4 查询虚拟接口列表

功能介绍

查询租户所有的虚拟接口列表

调用方法

请参见[如何调用API](#)。

URI

GET /v3/{project_id}/dcaas/virtual-interfaces

表 5-100 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	租户项目ID

表 5-101 Query 参数

参数	是否必选	参数类型	描述
limit	否	Integer	每页返回的个数。取值范围： 1~2000。 最小值：1 最大值：2000 缺省值：2000
marker	否	String	上一页最后一条资源记录的ID， 为空时为查询第一页。使用说明： 必须与limit一起使用。 最小长度：0 最大长度：36
fields	否	Array	显示字段列表 数组长度：1 - 5
sort_dir	否	Array	返回结果按照升序(asc)或降序 (desc)排列，默认为asc
sort_key	否	String	排序字段。 缺省值：id 最小长度：0 最大长度：36
enterprise_project_id	否	Array	根据企业项目ID过滤资源实例 数组长度：1 - 10
id	否	Array	根据资源ID过滤实例 数组长度：1 - 5
status	否	Array	根据资源状态过滤实例 数组长度：1 - 5
direct_connect_id	否	Array	根据物理专线ID过滤查询实例信息 数组长度：1 - 5

参数	是否必选	参数类型	描述
vgw_id	否	Array	根据虚拟网关ID过滤查询实例信息 数组长度：1 - 5

请求参数

表 5-102 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。获取Token，请参考《统一身份认证服务API参考》的“获取用户Token”章节。请求响应成功后在响应消息头中包含的“X-Subject-Token”的值即为Token值。 最小长度：0 最大长度：10240

响应参数

状态码：200

表 5-103 响应 Body 参数

参数	参数类型	描述
request_id	String	操作请求ID
virtual_interfaces	Array of VirtualInterface objects	虚拟接口对象
page_info	PageInfo object	分页查询页的信息

表 5-104 VirtualInterface

参数	参数类型	描述
id	String	虚拟接口的ID 最大长度：36

参数	参数类型	描述
name	String	虚拟接口的名字 最大长度：64
admin_state_up	Boolean	管理状态：true或false
bandwidth	Integer	虚拟接口接入带宽 最小值：2 最大值：2147483647
create_time	String	虚拟接口创建时间。采用UTC时间格式，格式为：yyyy-MM-ddTHH:mm:ss.SSSZ 最大长度：255
description	String	虚拟接口的描述 最大长度：128
direct_connect_id	String	物理专线的ID 最大长度：36
service_type	String	接入网关的类型：包括VGW、GDGW、LGW等枚举值： <ul style="list-style-type: none">• VGW• GDGW• LGW
status	String	操作状态，合法值是：ACTIVE, DOWN, BUILD, ERROR, PENDING_CREATE, PENDING_UPDATE, PENDING_DELETE, DELETED, AUTHORIZATION, REJECTED
tenant_id	String	租户ID 最小长度：32 最大长度：32
type	String	表示接口类型：private 缺省值：private 最大长度：255 枚举值： <ul style="list-style-type: none">• private• public
vgw_id	String	虚拟网关的ID 最小长度：36 最大长度：36

参数	参数类型	描述
vlan	Integer	同用户网关对接的vlan, 配置范围0-3999 最小值: 0 最大值: 3999
route_limit	Integer	VIF远端子网路由配置规格 最小值: 1 最大值: 200 缺省值: 50
enable_nqa	Boolean	是否使能nqa功能: true或false
enable_bfd	Boolean	是否使能bfd功能: true或false
lag_id	String	VIF关联的链路聚合组ID 最小长度: 36 最大长度: 36
device_id	String	归属的设备ID
enterprise_project_id	String	实例所属企业项目ID 最小长度: 36 最大长度: 36
tags	Array of Tag objects	标签信息 数组长度: 0 - 10
local_gateway_v4_ip	String	云侧网关IPv4接口地址, 该字段现已经移到vifpeer参数列表中, 未来将会废弃。
remote_gateway_v4_ip	String	客户侧网关IPv4接口地址, 该字段现已经移到vifpeer参数列表中, 未来将会废弃。
ies_id	String	归属的IES站点的ID
reason	String	如果资源的状态是Error的情况下, 该参数会显示相关错误信息。
rate_limit	Boolean	标识虚拟接口是否开启限速
address_family	String	接口的地址簇类型, ipv4, ipv6。该字段现已迁移到vifpeer参数列表中, 未来将会废弃。
local_gateway_v6_ip	String	云侧网关IPv6接口地址, 该字段现已迁移到vifpeer参数列表中, 未来将会废弃。
remote_gateway_v6_ip	String	客户侧网关IPv6接口地址, 该字段现已迁移到vifpeer参数列表中, 未来将会废弃。
lgw_id	String	本地网关的ID, 用于IES场景。
gateway_id	String	虚拟接口关联的网关的ID

参数	参数类型	描述
remote_ep_group	Array of strings	远端子网列表，记录租户侧的cidrs。该字段现已迁移到vifpeer参数列表中，未来将会废弃。
service_ep_group	Array of strings	该字段用于公网专线接口，表示租户可以访问云上公网服务地址列表。该字段现已迁移到vifpeer参数列表中，未来将会废弃。
bgp_route_limit	Integer	BGP的路由配置规格
priority	String	虚拟接口的优先级，支持两种优先级状态normal和low。接口优先级相同时表示负载关系，接口优先级不同时表示主备关系，出云流量优先转到优先级更高的normal接口。目前仅BGP模式接口支持。 缺省值： normal 枚举值： <ul style="list-style-type: none">• normal• low
vif_peers	Array of VifPeer objects	vif的Peer的相关信息
extend_attribute	VifExtendAttribute object	扩展参数信息

表 5-105 Tag

参数	参数类型	描述
key	String	标签键，最大长度36个unicode字符，格式为大小写字母，数字，中划线“-”，下划线“_”，中文。 最小长度： 0 最大长度： 36
value	String	标签值，最大长度43个unicode字符，格式为大小写字母，数字，中划线“-”，下划线“_”，点“.”，中文。 最小长度： 0 最大长度： 43

表 5-106 VifPeer

参数	参数类型	描述
id	String	资源ID 最小长度：36 最大长度：36
tenant_id	String	归属租户ID 最小长度：36 最大长度：36
name	String	VIF对等体名字 最小长度：0 最大长度：64
description	String	VIF对等体名字描述信息 最小长度：0 最大长度：128
address_family	String	接口的地址簇类型， ipv4, ipv6
local_gateway_ip	String	VIF对等体云侧接口地址
remote_gateway_ip	String	VIF对等体客户侧接口地址
route_mode	String	路由模式： static/bgp 最大长度：255 枚举值： <ul style="list-style-type: none">• bgp• static
bgp_asn	Integer	BGP邻居的AS号 最小值：1 最大值：4294967295
bgp_md5	String	BGP邻居的MD5密码
remote_ep_group	Array of strings	远端子网列表，记录租户侧的cidrs
service_ep_group	Array of strings	该字段用于公网专线接口,表示租户可以访问云上公网服务地址列表
device_id	String	归属的设备ID
bgp_route_limit	Integer	BGP的路由配置规格

参数	参数类型	描述
bgp_status	String	接口BGP协议状态,如果是静态路由接口则状态为null 最大长度: 10
status	String	VIF对等体状态
vif_id	String	vif对等体对应的虚拟接口ID 最小长度: 36 最大长度: 36
receive_route_num	Integer	路由模式为bgp: receive_route_num值为接收搭配BGP的路由数目; 路由模式为static: 该字段无意义, 值为-1; 注: 若早期接入云上的部分用户无法获取该字段信息, 如需要请联系客服迁移专线端口。
enable_nqa	Boolean	是否使能nqa功能: true或false
enable_bfd	Boolean	是否使能bfd功能: true或false

表 5-107 VifExtendAttribute

参数	参数类型	描述
ha_type	String	虚拟接口的可用性检测类型 枚举值: <ul style="list-style-type: none">• nqa• bfd
ha_mode	String	检测的具体的配置模式 枚举值: <ul style="list-style-type: none">• auto_single• auto_multi• static_single• static_multi• enhance_nqa
detect_multiplier	Integer	检测的重试次数 缺省值: 5
min_rx_interval	Integer	检测的接收时长间隔 缺省值: 1000
min_tx_interval	Integer	检测的发送时长间隔 缺省值: 1000

参数	参数类型	描述
remote_disclaim	Integer	检测的远端的标识，用于静态BFD
local_disclaim	Integer	检测的本端的标识，用于静态BFD

表 5-108 PageInfo

参数	参数类型	描述
previous_marker	String	上一页的marker，值为资源的uuid 最小长度：0 最大长度：36
current_count	Integer	当前列表中资源数量 最小值：0 最大值：2000
next_marker	String	下一页的marker，值为资源的uuid，为空时表示最后一页 最小长度：0 最大长度：36

请求示例

查询虚拟接口列表

```
GET https://{dc_endpoint}/v3/0605768a3300d5762f82c01180692873/dcaas/virtual-interfaces
```

响应示例

状态码：200

OK

```
{
  "virtual_interfaces": [ {
    "id": "0d0fdf63-f2c4-491c-8866-d504796189be",
    "name": "vif-0819",
    "description": "mytest",
    "tenant_id": "0605768a3300d5762f82c01180692873",
    "direct_connect_id": "4673e339-8412-4ee1-b73e-2ba9cdfa54c1",
    "vgw_id": "8a47064a-f34c-4f94-b7fe-cac456c9b37b",
    "type": "private",
    "service_type": "VGW",
    "vlan": 332,
    "bandwidth": 2,
    "status": "ACTIVE",
    "create_time": "2022-08-19T11:28:06.000Z",
    "admin_state_up": true,
    "enable_bfd": false,
    "route_limit": 50,
    "enable_nqa": false,
```

```
"local_gateway_v4_ip" : "1.1.1.1/30",
"remote_gateway_v4_ip" : "1.1.1.2/30",
"ies_id" : null,
"reason" : null,
"rate_limit" : false,
"address_family" : "ipv4",
"local_gateway_v6_ip" : null,
"remote_gateway_v6_ip" : null,
"lgw_id" : null,
"gateway_id" : null,
"remote_ep_group" : [ "1.1.2.0/30" ],
"service_ep_group" : [ ],
"bgp_route_limit" : 100,
"priority" : "normal",
"vif_peers" : [ {
  "id" : "c768eb52-12a8-4859-9b43-81194643040c",
  "tenant_id" : "0605768a3300d5762f82c01180692873",
  "name" : "vif-0819",
  "description" : "",
  "address_family" : "ipv4",
  "local_gateway_ip" : "1.1.1.1/30",
  "remote_gateway_ip" : "1.1.1.2/30",
  "route_mode" : "static",
  "bgp_asn" : null,
  "bgp_md5" : null,
  "device_id" : "18.9.215.131",
  "bgp_route_limit" : 100,
  "bgp_status" : null,
  "status" : "ACTIVE",
  "vif_id" : "0d0fdf63-f2c4-491c-8866-d504796189be",
  "receive_route_num" : -1,
  "remote_ep_group" : [ "1.1.2.0/30" ],
  "service_ep_group" : null,
  "enable_bfd" : false,
  "enable_nqa" : false
} ],
"enterprise_project_id" : "0"
} ],
"request_id" : "5633df7af874576d819a481c76673236"
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.dc.v3.region.DcRegion;
import com.huaweicloud.sdk.dc.v3.*;
import com.huaweicloud.sdk.dc.v3.model.*;

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

public class ListVirtualInterfacesSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
        // security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
        // environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running
```

this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment

```
String ak = System.getenv("CLOUD_SDK_AK");
String sk = System.getenv("CLOUD_SDK_SK");

ICredential auth = new BasicCredentials()
    .withAk(ak)
    .withSk(sk);

DcClient client = DcClient.newBuilder()
    .withCredential(auth)
    .withRegion(DcRegion.valueOf("<YOUR REGION>"))
    .build();
ListVirtualInterfacesRequest request = new ListVirtualInterfacesRequest();
request.withDirectConnectId();
request.withVgwId();
request.withLimit(<limit>);
request.withMarker("<marker>");
request.withFields();
request.withSortDir();
request.withSortKey("<sort_key>");
request.withEnterpriseProjectId();
request.withId();
request.withStatus();
try {
    ListVirtualInterfacesResponse response = client.listVirtualInterfaces(request);
    System.out.println(response.toString());
} catch (ConnectionException e) {
    e.printStackTrace();
} catch (RequestTimeoutException e) {
    e.printStackTrace();
} catch (ServiceResponseException e) {
    e.printStackTrace();
    System.out.println(e.getHttpStatusCode());
    System.out.println(e.getRequestId());
    System.out.println(e.getErrorCode());
    System.out.println(e.getErrorMsg());
}
}
```

Python

```
# coding: utf-8

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdc.v3.region.dc_region import DcRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdc.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    # risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    # variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this
    # example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.getenv("CLOUD_SDK_AK")
    sk = os.getenv("CLOUD_SDK_SK")

    credentials = BasicCredentials(ak, sk) \

    client = DcClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DcRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListVirtualInterfacesRequest()
        request.direct_connect_id =
        request.vgw_id =
```

```
request.limit = <limit>
request.marker = "<marker>"
request.fields =
request.sort_dir =
request.sort_key = "<sort_key>"
request.enterprise_project_id =
request.id =
request.status =
response = client.list_virtual_interfaces(request)
print(response)
except exceptions.ClientRequestException as e:
print(e.status_code)
print(e.request_id)
print(e.error_code)
print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dc "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    // risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    // variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this
    // example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := dc.NewDcClient(
        dc.DcClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ListVirtualInterfacesRequest{}
    limitRequest:= int32(<limit>)
    request.Limit = &limitRequest
    markerRequest:= "<marker>"
    request.Marker = &markerRequest
    sortKeyRequest:= "<sort_key>"
    request.SortKey = &sortKeyRequest
    response, err := client.ListVirtualInterfaces(request)
    if err == nil {
        fmt.Printf("%v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见[API Explorer](#)的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK

错误码

请参见[错误码](#)。

5.3.5 创建虚拟接口

功能介绍

虚拟接口配置物理专线上与客户互联的IP和路由等相关信息

调用方法

请参见[如何调用API](#)。

URI

POST /v3/{project_id}/dcaas/virtual-interfaces

表 5-109 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	租户项目ID

请求参数

表 5-110 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。获取Token，请参考《统一身份认证服务API参考》的“获取用户Token”章节。请求响应成功后在响应消息头中包含的“X-Subject-Token”的值即为Token值。 最小长度： 0 最大长度： 10240

表 5-111 请求 Body 参数

参数	是否必选	参数类型	描述
virtual_interface	是	CreateVirtualInterface object	创建虚拟接口对象参数

表 5-112 CreateVirtualInterface

参数	是否必选	参数类型	描述
name	否	String	虚拟接口名字 最小长度：0 最大长度：64
description	否	String	虚拟接口描述信息 最小长度：0 最大长度：128
direct_connect_id	否	String	虚拟接口关联的物理专线ID。在创建虚拟接口时，需要关联direct_connect_id或lag_id，当LAG功能在当前局点不支持时，该参数为必选。
type	是	String	虚拟接口的类型,private 枚举值： <ul style="list-style-type: none">• private• public
service_type	否	String	接入网关的类型：包括VGW、GDGW、LGW等 枚举值： <ul style="list-style-type: none">• VGW• GDGW• LGW
vlan	是	Integer	对接客户侧vlan。当关联物理连接选择托管专线时，vlan需要和托管专线保持一致。 最小值：0 最大值：3999
bandwidth	是	Integer	虚拟接口接入带宽 最小值：2 最大值：2147483647

参数	是否必选	参数类型	描述
local_gateway_v4_ip	否	String	云侧网关IPv4接口地址,如果address_family是IPv4, 是必选参数
remote_gateway_v4_ip	否	String	客户侧网关IPv4接口地址,如果address_family是IPv4, 是必选参数
address_family	否	String	接口的地址簇类型, ipv4, ipv6 缺省值: ipv4
local_gateway_v6_ip	否	String	云侧网关IPv6接口地址,如果address_family是IPv6, 是必选参数
remote_gateway_v6_ip	否	String	客户侧网关IPv6接口地址,如果address_family是IPv6, 是必选参数
vgw_id	是	String	虚拟接口连接的虚拟网关的ID
route_mode	是	String	路由模式: static/bgp 缺省值: static 枚举值: <ul style="list-style-type: none">• static• bgp
bgp_asn	否	Integer	客户侧BGP邻居的AS号 最小值: 1 最大值: 4294967295
bgp_md5	否	String	BGP邻居的MD5密码
remote_ep_group	是	Array of strings	远端子网列表, 记录租户侧的cidrs
service_ep_group	否	Array of strings	访问公网服务的子网列表
enable_bfd	否	Boolean	是否使能bfd功能: true或false。 缺省值: false
enable_nqa	否	Boolean	是否使能nqa功能: true或false。 缺省值: false
lag_id	否	String	虚拟接口关联的链路聚合组ID 最小长度: 36 最大长度: 36

参数	是否必选	参数类型	描述
resource_tenant_id	否	String	目标的租户的ID,用于跨租户创建虚拟接口场景
enterprise_project_id	否	String	实例所属企业项目ID 最小长度: 36 最大长度: 36
tags	否	Array of Tag objects	标签信息 数组长度: 0 - 10

表 5-113 Tag

参数	是否必选	参数类型	描述
key	是	String	标签键, 最大长度36个unicode字符, 格式为大小写字母, 数字, 中划线“-”, 下划线“_”, 中文。 最小长度: 0 最大长度: 36
value	否	String	标签值, 最大长度43个unicode字符, 格式为大小写字母, 数字, 中划线“-”, 下划线“_”, 点“.”, 中文。 最小长度: 0 最大长度: 43

响应参数

状态码: 201

表 5-114 响应 Body 参数

参数	参数类型	描述
request_id	String	操作请求ID
virtual_interface	VirtualInterface object	虚拟接口对象

表 5-115 VirtualInterface

参数	参数类型	描述
id	String	虚拟接口的ID 最大长度：36
name	String	虚拟接口的名字 最大长度：64
admin_state_up	Boolean	管理状态：true或false
bandwidth	Integer	虚拟接口接入带宽 最小值：2 最大值：2147483647
create_time	String	虚拟接口创建时间。采用UTC时间格式，格式为：yyyy-MM-ddTHH:mm:ss.SSSZ 最大长度：255
description	String	虚拟接口的描述 最大长度：128
direct_connect_id	String	物理专线的ID 最大长度：36
service_type	String	接入网关的类型：VGW 接入网关的类型：包括VGW、GDGW、LGW等 枚举值： <ul style="list-style-type: none">• VGW• GDGW• LGW
status	String	操作状态，合法值是：ACTIVE, DOWN, BUILD, ERROR, PENDING_CREATE, PENDING_UPDATE, PENDING_DELETE, DELETED, AUTHORIZATION, REJECTED
tenant_id	String	租户ID 最小长度：32 最大长度：32
type	String	表示接口类型：private 缺省值：private 最大长度：255 枚举值： <ul style="list-style-type: none">• private• public

参数	参数类型	描述
vgw_id	String	虚拟网关的ID 最小长度：36 最大长度：36
vlan	Integer	同用户网关对接的vlan, 配置范围0-3999 最小值：0 最大值：3999
route_limit	Integer	VIF远端子网路由配置规格 最小值：1 最大值：200 缺省值：50
enable_nqa	Boolean	是否使能nqa功能：true或false
enable_bfd	Boolean	是否使能bfd功能：true或false
lag_id	String	VIF关联的链路聚合组ID 最小长度：36 最大长度：36
device_id	String	归属的设备ID
enterprise_project_id	String	实例所属企业项目ID 最小长度：36 最大长度：36
tags	Array of Tag objects	标签信息 数组长度：0 - 10
local_gateway_v4_ip	String	云侧网关IPv4接口地址，该字段现已经移到vifpeer参数列表中，未来将会废弃。
remote_gateway_v4_ip	String	客户侧网关IPv4接口地址，该字段现已经移到vifpeer参数列表中，未来将会废弃。
ies_id	String	归属的IES站点的ID
reason	String	如果资源的状态是Error的情况下，该参数会显示相关错误信息。
rate_limit	Boolean	标识虚拟接口是否开启限速
address_family	String	接口的地址簇类型，ipv4，ipv6。该字段现已迁移到vifpeer参数列表中，未来将会废弃。
local_gateway_v6_ip	String	云侧网关IPv6接口地址，该字段现已迁移到vifpeer参数列表中，未来将会废弃。
remote_gateway_v6_ip	String	客户侧网关IPv6接口地址，该字段现已迁移到vifpeer参数列表中，未来将会废弃。

参数	参数类型	描述
lgw_id	String	本地网关的ID，用于IES场景。
gateway_id	String	虚拟接口关联的网关的ID
remote_ep_group	Array of strings	远端子网列表，记录租户侧的cidrs。该字段现已迁移到vifpeer参数列表中，未来将会废弃。
service_ep_group	Array of strings	该字段用于公网专线接口，表示租户可以访问云上公网服务地址列表。该字段现已迁移到vifpeer参数列表中，未来将会废弃。
bgp_route_limit	Integer	BGP的路由配置规格
priority	String	虚拟接口的优先级，支持两种优先级状态normal和low。接口优先级相同时表示负载关系，接口优先级不同时表示主备关系，出云流量优先转到优先级更高的normal接口。目前仅BGP模式接口支持。 缺省值： normal 枚举值： <ul style="list-style-type: none"> • normal • low
vif_peers	Array of VifPeer objects	vif的Peer的相关信息
extend_attribute	VifExtendAttribute object	扩展参数信息

表 5-116 Tag

参数	参数类型	描述
key	String	标签键，最大长度36个unicode字符，格式为大小写字母，数字，中划线“-”，下划线“_”，中文。 最小长度： 0 最大长度： 36
value	String	标签值，最大长度43个unicode字符，格式为大小写字母，数字，中划线“-”，下划线“_”，点“.”，中文。 最小长度： 0 最大长度： 43

表 5-117 VifPeer

参数	参数类型	描述
id	String	资源ID 最小长度：36 最大长度：36
tenant_id	String	归属租户ID 最小长度：36 最大长度：36
name	String	VIF对等体名字 最小长度：0 最大长度：64
description	String	VIF对等体名字描述信息 最小长度：0 最大长度：128
address_family	String	接口的地址簇类型， ipv4, ipv6
local_gateway_ip	String	VIF对等体云侧接口地址
remote_gateway_ip	String	VIF对等体客户侧接口地址
route_mode	String	路由模式： static/bgp 最大长度：255 枚举值： <ul style="list-style-type: none">• bgp• static
bgp_asn	Integer	BGP邻居的AS号 最小值：1 最大值：4294967295
bgp_md5	String	BGP邻居的MD5密码
remote_ep_group	Array of strings	远端子网列表，记录租户侧的cidrs
service_ep_group	Array of strings	该字段用于公网专线接口,表示租户可以访问云上公网服务地址列表
device_id	String	归属的设备ID
bgp_route_limit	Integer	BGP的路由配置规格

参数	参数类型	描述
bgp_status	String	接口BGP协议状态,如果是静态路由接口则状态为null 最大长度: 10
status	String	VIF对等体状态
vif_id	String	vif对等体对应的虚拟接口ID 最小长度: 36 最大长度: 36
receive_route_num	Integer	路由模式为bgp: receive_route_num值为接收搭配BGP的路由数目; 路由模式为static: 该字段无意义, 值为-1; 注: 若早期接入云上的部分用户无法获取该字段信息, 如需要请联系客服迁移专线端口。
enable_nqa	Boolean	是否使能nqa功能: true或false
enable_bfd	Boolean	是否使能bfd功能: true或false

表 5-118 VifExtendAttribute

参数	参数类型	描述
ha_type	String	虚拟接口的可用性检测类型 枚举值: <ul style="list-style-type: none">• nqa• bfd
ha_mode	String	检测的具体的配置模式 枚举值: <ul style="list-style-type: none">• auto_single• auto_multi• static_single• static_multi• enhance_nqa
detect_multiplier	Integer	检测的重试次数 缺省值: 5
min_rx_interval	Integer	检测的接收时长间隔 缺省值: 1000
min_tx_interval	Integer	检测的发送时长间隔 缺省值: 1000

参数	参数类型	描述
remote_disclaim	Integer	检测的远端的标识，用于静态BFD
local_disclaim	Integer	检测的本端的标识，用于静态BFD

请求示例

创建一个private类型的虚拟接口，设置带宽为2，VLAN为332，云侧网关IPv4接口地址为1.1.1.1/30，客户侧网关IPv4接口地址为1.1.1.2/30，使用静态路由模式。

```
POST https://{dc_endpoint}/v3/0605768a3300d5762f82c01180692873/dcaas/virtual-interfaces
```

```
{
  "virtual_interface": {
    "name": "vif-0819",
    "description": "mytest",
    "direct_connect_id": "4673e339-8412-4ee1-b73e-2ba9cdfa54c1",
    "vgw_id": "8a47064a-f34c-4f94-b7fe-cac456c9b37b",
    "vlan": 332,
    "bandwidth": 2,
    "local_gateway_v4_ip": "1.1.1.1/30",
    "remote_gateway_v4_ip": "1.1.1.2/30",
    "type": "private",
    "route_mode": "static",
    "remote_ep_group": [ "1.1.2.0/30" ]
  }
}
```

响应示例

状态码： 201

Created

```
{
  "virtual_interface": {
    "id": "0d0fdf63-f2c4-491c-8866-d504796189be",
    "name": "vif-0819",
    "description": "mytest",
    "tenant_id": "0605768a3300d5762f82c01180692873",
    "direct_connect_id": "4673e339-8412-4ee1-b73e-2ba9cdfa54c1",
    "vgw_id": "8a47064a-f34c-4f94-b7fe-cac456c9b37b",
    "type": "private",
    "service_type": "VGW",
    "vlan": 332,
    "bandwidth": 2,
    "device_id": "18.9.215.131",
    "status": "ACTIVE",
    "create_time": "2022-08-19T11:28:06.000Z",
    "admin_state_up": true,
    "enable_bfd": false,
    "route_limit": 50,
    "enable_nqa": false,
    "local_gateway_v4_ip": "1.1.1.1/30",
    "remote_gateway_v4_ip": "1.1.1.2/30",
    "ies_id": null,
    "reason": null,
    "rate_limit": false,
    "local_gateway_v6_ip": null,
    "remote_gateway_v6_ip": null,
    "lgw_id": null,
  }
}
```

```
"lag_id" : null,
"priority" : "normal",
"vif_peers" : [ {
  "id" : "c768eb52-12a8-4859-9b43-81194643040c",
  "tenant_id" : "0605768a3300d5762f82c01180692873",
  "name" : "vif-0819",
  "description" : "",
  "address_family" : "ipv4",
  "local_gateway_ip" : "1.1.1.1/30",
  "remote_gateway_ip" : "1.1.1.2/30",
  "route_mode" : "static",
  "bgp_asn" : null,
  "bgp_md5" : null,
  "device_id" : "18.9.215.131",
  "bgp_route_limit" : 100,
  "bgp_status" : null,
  "status" : "ACTIVE",
  "vif_id" : "0d0fdf63-f2c4-491c-8866-d504796189be",
  "receive_route_num" : -1,
  "remote_ep_group" : [ "1.1.2.0/30" ],
  "enable_bfd" : false,
  "enable_nqa" : false,
  "service_ep_group" : null
} ],
"enterprise_project_id" : "0"
},
"request_id" : "5633df7af874576d819a481c76673236"
}
```

SDK 代码示例

SDK代码示例如下。

Java

创建一个private类型的虚拟接口，设置带宽为2，VLAN为332，云侧网关IPv4接口地址为1.1.1.1/30，客户侧网关IPv4接口地址为1.1.1.2/30，使用静态路由模式。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.dc.v3.region.DcRegion;
import com.huaweicloud.sdk.dc.v3.*;
import com.huaweicloud.sdk.dc.v3.model.*;

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

public class CreateVirtualInterfaceSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
        // security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
        // environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running
        // this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
            .withAk(ak)
            .withSk(sk);
```



```
DcClient client = DcClient.newBuilder()
    .withCredential(auth)
    .withRegion(DcRegion.valueOf("<YOUR REGION>"))
    .build();
CreateVirtualInterfaceRequest request = new CreateVirtualInterfaceRequest();
CreateVirtualInterfaceRequestBody body = new CreateVirtualInterfaceRequestBody();
List<String> listVirtualInterfaceRemoteEpGroup = new ArrayList<>();
listVirtualInterfaceRemoteEpGroup.add("1.1.2.0/30");
CreateVirtualInterface virtualInterfacebody = new CreateVirtualInterface();
virtualInterfacebody.withName("vif-0819")
    .withDescription("mytest")
    .withDirectConnectId("4673e339-8412-4ee1-b73e-2ba9cdfa54c1")
    .withType(CreateVirtualInterface.TypeEnum.fromValue("private"))
    .withVlan(332)
    .withBandwidth(2)
    .withLocalGatewayV4Ip("1.1.1.1/30")
    .withRemoteGatewayV4Ip("1.1.1.2/30")
    .withVgwId("8a47064a-f34c-4f94-b7fe-cac456c9b37b")
    .withRouteMode(CreateVirtualInterface.RouteModeEnum.fromValue("static"))
    .withRemoteEpGroup(listVirtualInterfaceRemoteEpGroup);
body.withVirtualInterface(virtualInterfacebody);
request.withBody(body);
try {
    CreateVirtualInterfaceResponse response = client.createVirtualInterface(request);
    System.out.println(response.toString());
} catch (ConnectionException e) {
    e.printStackTrace();
} catch (RequestTimeoutException e) {
    e.printStackTrace();
} catch (ServiceResponseException e) {
    e.printStackTrace();
    System.out.println(e.getStatusCode());
    System.out.println(e.getRequestId());
    System.out.println(e.getErrorCode());
    System.out.println(e.getErrorMsg());
}
}
```

Python

创建一个private类型的虚拟接口，设置带宽为2，VLAN为332，云侧网关IPv4接口地址为1.1.1.1/30，客户侧网关IPv4接口地址为1.1.1.2/30，使用静态路由模式。

```
# coding: utf-8

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdc.v3.region.dc_region import DcRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdc.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    # risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    # variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this
    # example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.getenv("CLOUD_SDK_AK")
    sk = os.getenv("CLOUD_SDK_SK")

    credentials = BasicCredentials(ak, sk) \

    client = DcClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DcRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = CreateVirtualInterfaceRequest()
```

```
listRemoteEpGroupVirtualInterface = [
    "1.1.2.0/30"
]
virtualInterfacebody = CreateVirtualInterface(
    name="vif-0819",
    description="mytest",
    direct_connect_id="4673e339-8412-4ee1-b73e-2ba9cdfa54c1",
    type="private",
    vlan=332,
    bandwidth=2,
    local_gateway_v4_ip="1.1.1.1/30",
    remote_gateway_v4_ip="1.1.1.2/30",
    vgw_id="8a47064a-f34c-4f94-b7fe-cac456c9b37b",
    route_mode="static",
    remote_ep_group=listRemoteEpGroupVirtualInterface
)
request.body = CreateVirtualInterfaceRequestBody(
    virtual_interface=virtualInterfacebody
)
response = client.create_virtual_interface(request)
print(response)
except exceptions.ClientRequestException as e:
    print(e.status_code)
    print(e.request_id)
    print(e.error_code)
    print(e.error_msg)
```

Go

创建一个private类型的虚拟接口，设置带宽为2，VLAN为332，云侧网关IPv4接口地址为1.1.1.1/30，客户侧网关IPv4接口地址为1.1.1.2/30，使用静态路由模式。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dc "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    // risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    // variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this
    // example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := dc.NewDcClient(
        dc.DcClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.CreateVirtualInterfaceRequest{}
    var listRemoteEpGroupVirtualInterface = []string{
        "1.1.2.0/30",
    }
    nameVirtualInterface:= "vif-0819"
    descriptionVirtualInterface:= "mytest"
    directConnectIdVirtualInterface:= "4673e339-8412-4ee1-b73e-2ba9cdfa54c1"
```

```
localGatewayV4IpVirtualInterface:= "1.1.1.1/30"
remoteGatewayV4IpVirtualInterface:= "1.1.1.2/30"
virtualInterfacebody := &model.CreateVirtualInterface{
    Name: &nameVirtualInterface,
    Description: &descriptionVirtualInterface,
    DirectConnectId: &directConnectIdVirtualInterface,
    Type: model.GetCreateVirtualInterfaceTypeEnum().PRIVATE,
    Vlan: int32(332),
    Bandwidth: int32(2),
    LocalGatewayV4Ip: &localGatewayV4IpVirtualInterface,
    RemoteGatewayV4Ip: &remoteGatewayV4IpVirtualInterface,
    VgwId: "8a47064a-f34c-4f94-b7fe-cac456c9b37b",
    RouteMode: model.GetCreateVirtualInterfaceRouteModeEnum().STATIC,
    RemoteEpGroup: listRemoteEpGroupVirtualInterface,
}
request.Body = &model.CreateVirtualInterfaceRequestBody{
    VirtualInterface: virtualInterfacebody,
}
response, err := client.CreateVirtualInterface(request)
if err == nil {
    fmt.Printf("%+v\n", response)
} else {
    fmt.Println(err)
}
}
```

更多

更多编程语言的SDK代码示例，请参见[API Explorer](#)的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
201	Created

错误码

请参见[错误码](#)。

5.3.6 更新虚拟接口对等体

功能介绍

更新虚拟接口对等体信息,包括远端子网,名字和描述等。本接口只在支持IPv6的区域开放,如需要使用请联系客服

调用方法

请参见[如何调用API](#)。

URI

PUT /v3/{project_id}/dcaas/vif-peers/{vif_peer_id}

表 5-119 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	租户项目ID
vif_peer_id	是	String	虚拟接口对等体ID

请求参数

表 5-120 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。获取Token，请参考《统一身份认证服务API参考》的“获取用户Token”章节。请求响应成功后在响应消息头中包含的“X-Subject-Token”的值即为Token值。 最小长度：0 最大长度：10240

表 5-121 请求 Body 参数

参数	是否必选	参数类型	描述
vif_peer	否	UpdateVifPeer object	更新虚拟接口对等体参数

表 5-122 UpdateVifPeer

参数	是否必选	参数类型	描述
name	否	String	VIF对等体名字 最小长度：0 最大长度：64
description	否	String	VIF对等体名字描述信息 最小长度：0 最大长度：128
remote_ep_group	否	Array of strings	远端子网列表，记录用户侧的cidrs

响应参数

状态码：200

表 5-123 响应 Body 参数

参数	参数类型	描述
vif_peer	VifPeer object	虚拟接口对等体对象

表 5-124 VifPeer

参数	参数类型	描述
id	String	资源ID 最小长度：36 最大长度：36
tenant_id	String	归属租户ID 最小长度：36 最大长度：36
name	String	VIF对等体名字 最小长度：0 最大长度：64
description	String	VIF对等体名字描述信息 最小长度：0 最大长度：128
address_family	String	接口的地址簇类型，ipv4，ipv6
local_gateway_ip	String	VIF对等体云侧接口地址
remote_gateway_ip	String	VIF对等体客户侧接口地址
route_mode	String	路由模式：static/bgp 最大长度：255 枚举值： <ul style="list-style-type: none">• bgp• static
bgp_asn	Integer	BGP邻居的AS号 最小值：1 最大值：4294967295

参数	参数类型	描述
bgp_md5	String	BGP邻居的MD5密码
remote_ep_group	Array of strings	远端子网列表，记录租户侧的cidrs
service_ep_group	Array of strings	该字段用于公网专线接口,表示租户可以访问云上公网服务地址列表
device_id	String	归属的设备ID
bgp_route_limit	Integer	BGP的路由配置规格
bgp_status	String	接口BGP协议状态,如果是静态路由接口则状态为null 最大长度：10
status	String	VIF对等体状态
vif_id	String	vif对等体对应的虚拟接口ID 最小长度：36 最大长度：36
receive_route_num	Integer	路由模式为bgp: receive_route_num值为接收搭配BGP的路由数目；路由模式为static: 该字段无意义，值为-1；注：若早期接入云上的部分用户无法获取该字段信息，如需要请联系客服迁移专线端口。
enable_nqa	Boolean	是否使能nqa功能：true或false
enable_bfd	Boolean	是否使能bfd功能：true或false

请求示例

更新虚拟接口对等体的名称和名称，修改远端子网列表。

```
PUT https://{endpoint}/v3/cfa563efb77d4b6d9960781d82530fd8/dcaas/vif-peers/68250543-0a13-4ac7-aa36-d018856ac640
```

```
{
  "vif_peer": {
    "name": "vif-0819",
    "description": "mytest",
    "remote_ep_group": [ "20.1.1.0/24" ]
  }
}
```

响应示例

状态码：200

OK

```
{
  "vif_peer": {
```

```
"name" : "vif-0819",
"description" : "mytest",
"id" : "4c95de3e-9f75-4357-9c79-b22498dd71c7",
"tenant_id" : "ed28c294165741faeccab26913122a1",
"address_family" : "ipv4",
"local_gateway_ip" : "12.3.4.1/30",
"remote_gateway_ip" : "12.3.4.2/30",
"route_mode" : "static",
"bgp_asn" : null,
"bgp_md5" : null,
"bgp_route_limit" : 100,
"bgp_status" : null,
"status" : "ACTIVE",
"vif_id" : "5d6c17bc-0ebe-420b-8734-21f519e9d7ad",
"receive_route_num" : -1,
"remote_ep_group" : [ "20.1.1.0/24" ]
}
}
```

SDK 代码示例

SDK代码示例如下。

Java

更新虚拟接口对等体的名称和名称，修改远端子网列表。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.dc.v3.region.DcRegion;
import com.huaweicloud.sdk.dc.v3.*;
import com.huaweicloud.sdk.dc.v3.model.*;

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

public class UpdateVifPeerSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
        // security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
        // environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running
        // this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
            .withAk(ak)
            .withSk(sk);

        DcClient client = DcClient.newBuilder()
            .withCredential(auth)
            .withRegion(DcRegion.valueOf("<YOUR REGION>"))
            .build();
        UpdateVifPeerRequest request = new UpdateVifPeerRequest();
        UpdateVifPeerRequestBody body = new UpdateVifPeerRequestBody();
        List<String> listVifPeerRemoteEpGroup = new ArrayList<>();
        listVifPeerRemoteEpGroup.add("20.1.1.0/24");
        UpdateVifPeer vifPeerbody = new UpdateVifPeer();
        vifPeerbody.withName("vif-0819")
            .withDescription("mytest")
    }
}
```

```
.withRemoteEpGroup(listVifPeerRemoteEpGroup);
body.withVifPeer(vifPeerbody);
request.withBody(body);
try {
    UpdateVifPeerResponse response = client.updateVifPeer(request);
    System.out.println(response.toString());
} catch (ConnectionException e) {
    e.printStackTrace();
} catch (RequestTimeoutException e) {
    e.printStackTrace();
} catch (ServiceResponseException e) {
    e.printStackTrace();
    System.out.println(e.getHttpStatusCode());
    System.out.println(e.getRequestId());
    System.out.println(e.getErrorCode());
    System.out.println(e.getErrorMsg());
}
}
```

Python

更新虚拟接口对等体的名称和名称，修改远端子网列表。

```
# coding: utf-8

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdc.v3.region.dc_region import DcRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdc.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this
    example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.getenv("CLOUD_SDK_AK")
    sk = os.getenv("CLOUD_SDK_SK")

    credentials = BasicCredentials(ak, sk) \

    client = DcClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DcRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = UpdateVifPeerRequest()
        listRemoteEpGroupVifPeer = [
            "20.1.1.0/24"
        ]
        vifPeerbody = UpdateVifPeer(
            name="vif-0819",
            description="mytest",
            remote_ep_group=listRemoteEpGroupVifPeer
        )
        request.body = UpdateVifPeerRequestBody(
            vif_peer=vifPeerbody
        )
        response = client.update_vif_peer(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```


Go

更新虚拟接口对等体的名称和名称，修改远端子网列表。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dc "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    // risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    // variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this
    // example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := dc.NewDcClient(
        dc.DcClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.UpdateVifPeerRequest{}
    var listRemoteEpGroupVifPeer = []string{
        "20.1.1.0/24",
    }
    nameVifPeer := "vif-0819"
    descriptionVifPeer := "mytest"
    vifPeerbody := &model.UpdateVifPeer{
        Name: &nameVifPeer,
        Description: &descriptionVifPeer,
        RemoteEpGroup: &listRemoteEpGroupVifPeer,
    }
    request.Body = &model.UpdateVifPeerRequestBody{
        VifPeer: vifPeerbody,
    }
    response, err := client.UpdateVifPeer(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见[API Explorer](#)的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK

错误码

请参见[错误码](#)。

5.3.7 删除虚拟接口对应的对等体

功能介绍

删除虚拟接口对等体信息,虚拟接口至少要含一个对等体,最后一个对等体不能删除 本接口只在支持Ipv6的区域开放,如需要使用请联系客服

调用方法

请参见[如何调用API](#)。

URI

DELETE /v3/{project_id}/dcaas/vif-peers/{vif_peer_id}

表 5-125 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	租户项目ID
vif_peer_id	是	String	虚拟接口对等体ID

请求参数

表 5-126 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。获取Token,请参考《统一身份认证服务API参考》的“获取用户Token”章节。请求响应成功后在响应消息头中包含的“X-Subject-Token”的值即为Token值。 最小长度: 0 最大长度: 10240

响应参数

无

请求示例

删除虚拟接口对等体

```
DELETE https://{endpoint}/v3/cfa563efb77d4b6d9960781d82530fd8/dcaas/vif-peers/68250543-0a13-4ac7-aa36-d018856ac640
```

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.dc.v3.region.DcRegion;
import com.huaweicloud.sdk.dc.v3.*;
import com.huaweicloud.sdk.dc.v3.model.*;

public class DeleteVifPeerSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
        // security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
        // environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running
        // this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
            .withAk(ak)
            .withSk(sk);

        DcClient client = DcClient.newBuilder()
            .withCredential(auth)
            .withRegion(DcRegion.valueOf("<YOUR REGION>"))
            .build();
        DeleteVifPeerRequest request = new DeleteVifPeerRequest();
        try {
            DeleteVifPeerResponse response = client.deleteVifPeer(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
        }
    }
}
```

```
        System.out.println(e.getErrorMsg());
    }
}
}
```

Python

```
# coding: utf-8

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdc.v3.region.dc_region import DcRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdc.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    # risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    # variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this
    # example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.getenv("CLOUD_SDK_AK")
    sk = os.getenv("CLOUD_SDK_SK")

    credentials = BasicCredentials(ak, sk) \

    client = DcClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DcRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = DeleteVifPeerRequest()
        response = client.delete_vif_peer(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dc "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    // risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    // variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this
    // example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := dc.NewDcClient(
        dc.DcClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
```

```
WithCredential(auth).  
Build()  
  
request := &model.DeleteVifPeerRequest{}  
response, err := client.DeleteVifPeer(request)  
if err == nil {  
    fmt.Printf("%+v\n", response)  
} else {  
    fmt.Println(err)  
}  
}
```

更多

更多编程语言的SDK代码示例，请参见[API Explorer](#)的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
204	No Content

错误码

请参见[错误码](#)。

5.3.8 创建虚拟接口对等体

功能介绍

每个虚拟接口可支持两个对等体，IPv4和IPv6对等体，在创建虚拟接口时默认创建IPv4对等体，本接口一般用于增加ipv6对等体。创建虚拟接口对接体之后，可以通过虚拟接口查询配置结果 本接口只在支持Ipv6的区域开放，如需要使用请联系客服

调用方法

请参见[如何调用API](#)。

URI

POST /v3/{project_id}/dcaas/vif-peers

表 5-127 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	租户项目ID

请求参数

表 5-128 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。获取Token，请参考《统一身份认证服务API参考》的“获取用户Token”章节。请求响应成功后在响应消息头中包含的“X-Subject-Token”的值即为Token值。 最小长度：0 最大长度：10240

表 5-129 请求 Body 参数

参数	是否必选	参数类型	描述
vif_peer	否	CreateVifPeer object	创建虚拟接口对等体参数

表 5-130 CreateVifPeer

参数	是否必选	参数类型	描述
name	否	String	VIF对等体名字 最小长度：0 最大长度：64
description	否	String	VIF对等体名字描述信息 最小长度：0 最大长度：128
address_family	否	String	接口的地址簇类型，ipv4，ipv6
local_gateway_ip	否	String	VIF对等体云侧接口地址
remote_gateway_ip	否	String	VIF对等体客户侧接口地址
route_mode	否	String	路由模式：static/bgp 最大长度：255 枚举值： <ul style="list-style-type: none">• bgp• static

参数	是否必选	参数类型	描述
bgp_asn	否	Integer	BGP邻居的AS号 最小值：1 最大值：4294967295
bgp_md5	否	String	BGP邻居的MD5密码
remote_ep_group	否	Array of strings	远端子网列表，记录租户侧的cidrs
vif_id	否	String	vif对等体对应的虚拟接口ID 最小长度：36 最大长度：36

响应参数

状态码：201

表 5-131 响应 Body 参数

参数	参数类型	描述
vif_peer	VifPeer object	虚拟接口对等体对象

表 5-132 VifPeer

参数	参数类型	描述
id	String	资源ID 最小长度：36 最大长度：36
tenant_id	String	归属租户ID 最小长度：36 最大长度：36
name	String	VIF对等体名字 最小长度：0 最大长度：64
description	String	VIF对等体名字描述信息 最小长度：0 最大长度：128

参数	参数类型	描述
address_family	String	接口的地址簇类型, ipv4, ipv6
local_gateway_ip	String	VIF对等体云侧接口地址
remote_gateway_ip	String	VIF对等体客户侧接口地址
route_mode	String	路由模式: static/bgp 最大长度: 255 枚举值: <ul style="list-style-type: none">• bgp• static
bgp_asn	Integer	BGP邻居的AS号 最小值: 1 最大值: 4294967295
bgp_md5	String	BGP邻居的MD5密码
remote_ep_group	Array of strings	远端子网列表, 记录租户侧的cidrs
service_ep_group	Array of strings	该字段用于公网专线接口,表示租户可以访问云上公网服务地址列表
device_id	String	归属的设备ID
bgp_route_limit	Integer	BGP的路由配置规格
bgp_status	String	接口BGP协议状态,如果是静态路由接口则状态为null 最大长度: 10
status	String	VIF对等体状态
vif_id	String	vif对等体对应的虚拟接口ID 最小长度: 36 最大长度: 36
receive_route_num	Integer	路由模式为bgp: receive_route_num值为接收搭配BGP的路由数目; 路由模式为static: 该字段无意义, 值为-1; 注: 若早期接入云上的部分用户无法获取该字段信息, 如需要请联系客服迁移专线端口。
enable_nqa	Boolean	是否使能nqa功能: true或false
enable_bfd	Boolean	是否使能bfd功能: true或false

请求示例

创建一个虚拟接口对等体，远端子网列表为20.1.1.0/24。

```
POST https://{endpoint}/v3/cfa563efb77d4b6d9960781d82530fd8/dcaas/vif-peers/68250543-0a13-4ac7-aa36-d018856ac640
```

```
{
  "vif_peer" : {
    "name" : "vif-0819",
    "description" : "mytest",
    "remote_ep_group" : [ "20.1.1.0/24" ]
  }
}
```

响应示例

状态码： 201

Created

```
{
  "vif_peer" : {
    "name" : "vif-0819",
    "description" : "mytest",
    "id" : "4c95de3e-9f75-4357-9c79-b22498dd71c7",
    "tenant_id" : "ed28c294165741faeaccab26913122a1",
    "address_family" : "ipv4",
    "local_gateway_ip" : "12.3.4.1/30",
    "remote_gateway_ip" : "12.3.4.2/30",
    "route_mode" : "static",
    "bgp_asn" : null,
    "bgp_md5" : null,
    "bgp_route_limit" : 100,
    "bgp_status" : null,
    "status" : "ACTIVE",
    "vif_id" : "5d6c17bc-0ebe-420b-8734-21f519e9d7ad",
    "receive_route_num" : -1,
    "remote_ep_group" : [ "12.3.4.0/30 172.56.0.0/16" ]
  }
}
```

SDK 代码示例

SDK代码示例如下。

Java

创建一个虚拟接口对等体，远端子网列表为20.1.1.0/24。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.dc.v3.region.DcRegion;
import com.huaweicloud.sdk.dc.v3.*;
import com.huaweicloud.sdk.dc.v3.model.*;

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

public class CreateVifPeerSolution {
```

```
public static void main(String[] args) {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
    // security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
    // environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running
    // this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    String ak = System.getenv("CLOUD_SDK_AK");
    String sk = System.getenv("CLOUD_SDK_SK");

    ICredential auth = new BasicCredentials()
        .withAk(ak)
        .withSk(sk);

    DcClient client = DcClient.newBuilder()
        .withCredential(auth)
        .withRegion(DcRegion.valueOf("<YOUR REGION>"))
        .build();
    CreateVifPeerRequest request = new CreateVifPeerRequest();
    CreateVifPeerRequestBody body = new CreateVifPeerRequestBody();
    List<String> listVifPeerRemoteEpGroup = new ArrayList<>();
    listVifPeerRemoteEpGroup.add("20.1.1.0/24");
    CreateVifPeer vifPeerbody = new CreateVifPeer();
    vifPeerbody.withName("vif-0819")
        .withDescription("mytest")
        .withRemoteEpGroup(listVifPeerRemoteEpGroup);
    body.withVifPeer(vifPeerbody);
    request.withBody(body);
    try {
        CreateVifPeerResponse response = client.createVifPeer(request);
        System.out.println(response.toString());
    } catch (ConnectionException e) {
        e.printStackTrace();
    } catch (RequestTimeoutException e) {
        e.printStackTrace();
    } catch (ServiceResponseException e) {
        e.printStackTrace();
        System.out.println(e.getStatusCode());
        System.out.println(e.getRequestId());
        System.out.println(e.getErrorCode());
        System.out.println(e.getErrorMsg());
    }
}
```

Python

创建一个虚拟接口对等体，远端子网列表为20.1.1.0/24。

```
# coding: utf-8

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkcore.v3.region.dc_region import DcRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkcore.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    # risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    # variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this
    # example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.getenv("CLOUD_SDK_AK")
    sk = os.getenv("CLOUD_SDK_SK")

    credentials = BasicCredentials(ak, sk) \

    client = DcClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DcRegion.value_of("<YOUR REGION>")) \
```

```
.build()

try:
    request = CreateVifPeerRequest()
    listRemoteEpGroupVifPeer = [
        "20.1.1.0/24"
    ]
    vifPeerbody = CreateVifPeer(
        name="vif-0819",
        description="mytest",
        remote_ep_group=listRemoteEpGroupVifPeer
    )
    request.body = CreateVifPeerRequestBody(
        vif_peer=vifPeerbody
    )
    response = client.create_vif_peer(request)
    print(response)
except exceptions.ClientRequestException as e:
    print(e.status_code)
    print(e.request_id)
    print(e.error_code)
    print(e.error_msg)
```

Go

创建一个虚拟接口对等体，远端子网列表为20.1.1.0/24。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dc "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    // risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    // variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this
    // example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := dc.NewDcClient(
        dc.DcClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.CreateVifPeerRequest{}
    var listRemoteEpGroupVifPeer = []string{
        "20.1.1.0/24",
    }
    nameVifPeer:= "vif-0819"
    descriptionVifPeer:= "mytest"
    vifPeerbody := &model.CreateVifPeer{
        Name: &nameVifPeer,
        Description: &descriptionVifPeer,
        RemoteEpGroup: &listRemoteEpGroupVifPeer,
    }
    request.Body = &model.CreateVifPeerRequestBody{
```

```
VifPeer: vifPeerbody,
}
response, err := client.CreateVifPeer(request)
if err == nil {
    fmt.Printf("%+v\n", response)
} else {
    fmt.Println(err)
}
}
```

更多

更多编程语言的SDK代码示例，请参见[API Explorer](#)的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
201	Created

错误码

请参见[错误码](#)。

5.3.9 执行虚拟接口倒换测试

功能介绍

客户双专线接入，需要支持双线自动倒换，方便进行功能测试。虚拟接口进行倒换测试会导致接口关闭，业务流量中断。对于虚拟接口，支持“关闭接口”和“开放接口”两种操作。1、关闭接口：下发shutdown命令，关闭接口；2、开放接口：下发undo_shutdown命令，使能接口。倒换测试选择shutdown时，虚拟接口的状态为ADMIN_SHUTDOWN，此状态不允许虚拟接口的其他操作。倒换测试选择undo_shutdown时，虚拟接口的状态为ACTIVE。

调用方法

请参见[如何调用API](#)。

URI

POST /v3/{project_id}/dcaas/switchover-test

表 5-133 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	租户项目ID

请求参数

表 5-134 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。获取Token，请参考《统一身份认证服务API参考》的“获取用户Token”章节。请求响应成功后在响应消息头中包含的“X-Subject-Token”的值即为Token值。 最小长度：0 最大长度：10240

表 5-135 请求 Body 参数

参数	是否必选	参数类型	描述
switchover_test_record	否	CreateSwitchoverTest object	创建倒换测试记录对象参数

表 5-136 CreateSwitchoverTest

参数	是否必选	参数类型	描述
resource_id	是	String	倒换测试的资源对象ID
resource_type	否	String	倒换测试的资源对象类型 缺省值：virtual_interface 最小长度：0 最大长度：128 枚举值： <ul style="list-style-type: none">virtual_interface
operation	是	String	shutdown, undo_shutdown表示倒换测试操作类型 最小长度：0 最大长度：128 枚举值： <ul style="list-style-type: none">shutdownundo_shutdown

响应参数

状态码： 201

表 5-137 响应 Body 参数

参数	参数类型	描述
request_id	String	操作请求ID
switchover_test_record	SwitchoverTestRecord object	倒换测试记录信息

表 5-138 SwitchoverTestRecord

参数	参数类型	描述
id	String	倒换测试记录的唯一标识
tenant_id	String	租户ID 最小长度： 32 最大长度： 32
resource_id	String	倒换测试的资源对象ID
resource_type	String	倒换测试的资源对象类型 缺省值： virtual_interface 最小长度： 0 最大长度： 128 枚举值： • virtual_interface
operation	String	shutdown, undo_shutdown表示倒换测试操作类型 最小长度： 0 最大长度： 128 枚举值： • shutdown • undo_shutdown
start_time	String	倒换测试操作的开始时间
end_time	String	倒换测试操作的结束时间

参数	参数类型	描述
operate_statuses	String	倒换测试状态记录 STARTING: 初始状态 INPROGRESS: 配置下发中 COMPLETE: 配置下发完成 ERROR: 配置下发失败 枚举值: <ul style="list-style-type: none">● STARTING● INPROGRESS● COMPLETE● ERROR

请求示例

对虚拟接口下发shutdown命令，关闭接口。

```
POST https://{dc_endpoint}/v3/de58f033eb664102ba85e4a5db473ca5/dcaas/switchover-test
{
  "switchover_test_record": {
    "resource_type": "virtual_interface",
    "resource_id": "d0b3329c-0063-470c-b1dc-657656b2e540",
    "operation": "shutdown"
  }
}
```

响应示例

状态码： 201

Created

```
{
  "switchover_test_record": {
    "id": "862d61f1-d9ea-4093-ba0e-2b8d415e3ab3",
    "tenant_id": "de58f033eb664102ba85e4a5db473ca5",
    "resource_type": "virtual_interface",
    "resource_id": "d0b3329c-0063-470c-b1dc-657656b2e540",
    "operation": "shutdown",
    "start_time": "2023-10-09T18:41:23.000Z",
    "end_time": "2023-10-09T18:41:23.000Z",
    "operate_status": "STARTING"
  },
  "request_id": "f91634a12b116b6f946d7871f5b4de18"
}
```

SDK 代码示例

SDK代码示例如下。

Java

对虚拟接口下发shutdown命令，关闭接口。

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
```

```
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.dc.v3.region.DcRegion;
import com.huaweicloud.sdk.dc.v3.*;
import com.huaweicloud.sdk.dc.v3.model.*;

public class SwitchoverTestSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
        // security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
        // environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running
        // this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
            .withAk(ak)
            .withSk(sk);

        DcClient client = DcClient.newBuilder()
            .withCredential(auth)
            .withRegion(DcRegion.valueOf("<YOUR REGION>"))
            .build();
        SwitchoverTestRequest request = new SwitchoverTestRequest();
        CreateSwitchoverTestRequestBody body = new CreateSwitchoverTestRequestBody();
        CreateSwitchoverTest switchoverTestRecordbody = new CreateSwitchoverTest();
        switchoverTestRecordbody.withResourceId("d0b3329c-0063-470c-b1dc-657656b2e540")
            .withResourceType(CreateSwitchoverTest.ResourceTypeEnum.fromValue("virtual_interface"))
            .withOperation(CreateSwitchoverTest.OperationEnum.fromValue("shutdown"));
        body.withSwitchoverTestRecord(switchoverTestRecordbody);
        request.withBody(body);
        try {
            SwitchoverTestResponse response = client.switchoverTest(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

对虚拟接口下发shutdown命令，关闭接口。

```
# coding: utf-8

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdcdc.v3.region.dc_region import DcRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdcdc.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    # risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    # variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this
    # example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
```



```
ak = os.Getenv("CLOUD_SDK_AK")
sk = os.Getenv("CLOUD_SDK_SK")

credentials = BasicCredentials(ak, sk) \

client = DcClient.new_builder() \
    .with_credentials(credentials) \
    .with_region(DcRegion.value_of("<YOUR REGION>")) \
    .build()

try:
    request = SwitchoverTestRequest()
    switchoverTestRecordbody = CreateSwitchoverTest(
        resource_id="d0b3329c-0063-470c-b1dc-657656b2e540",
        resource_type="virtual_interface",
        operation="shutdown"
    )
    request.body = CreateSwitchoverTestRequestBody(
        switchover_test_record=switchoverTestRecordbody
    )
    response = client.switchover_test(request)
    print(response)
except exceptions.ClientRequestException as e:
    print(e.status_code)
    print(e.request_id)
    print(e.error_code)
    print(e.error_msg)
```

Go

对虚拟接口下发shutdown命令，关闭接口。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dc "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    // risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    // variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this
    // example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := dc.NewDcClient(
        dc.DcClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.SwitchoverTestRequest{
        resourceTypeSwitchoverTestRecord:=
model.GetCreateSwitchoverTestResourceTypeEnum().VIRTUAL_INTERFACE
        switchoverTestRecordbody := &model.CreateSwitchoverTest{
            ResourceId: "d0b3329c-0063-470c-b1dc-657656b2e540",
            ResourceType: &resourceTypeSwitchoverTestRecord,
            Operation: model.GetCreateSwitchoverTestOperationEnum().SHUTDOWN,
```

```
}
request.Body = &model.CreateSwitchoverTestRequestBody{
    SwitchoverTestRecord: switchoverTestRecordbody,
}
response, err := client.SwitchoverTest(request)
if err == nil {
    fmt.Printf("%+v\n", response)
} else {
    fmt.Println(err)
}
}
```

更多

更多编程语言的SDK代码示例，请参见[API Explorer](#)的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
201	Created

错误码

请参见[错误码](#)。

5.3.10 查询虚拟接口倒换测试记录列表

功能介绍

查询倒换测试记录列表，只展示operate_status为COMPELTE的记录。

调用方法

请参见[如何调用API](#)。

URI

GET /v3/{project_id}/dcaas/switchover-test

表 5-139 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	租户项目ID

表 5-140 Query 参数

参数	是否必选	参数类型	描述
limit	否	Integer	每页返回的个数。取值范围： 1~2000。 最小值： 1 最大值： 2000 缺省值： 2000
marker	否	String	上一页最后一条资源记录的ID， 为空时为查询第一页。使用说明： 必须与limit一起使用。 最小长度： 0 最大长度： 36
fields	否	Array	显示字段列表 数组长度： 1 - 5
sort_dir	否	Array	返回结果按照升序(asc)或降序 (desc)排列，默认为asc
sort_key	否	String	排序字段。 缺省值： id 最小长度： 0 最大长度： 36
resource_id	否	Array	通过RESOURCE-ID过滤倒换测 试记录信息

请求参数

表 5-141 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	是	String	用户Token。获取Token，请参 考《统一身份认证服务API参 考》的“获取用户Token”章 节。请求响应成功后在响应消息 头中包含的“X-Subject- Token”的值即为Token值。 最小长度： 0 最大长度： 10240

响应参数

状态码：**200**

表 5-142 响应 Body 参数

参数	参数类型	描述
request_id	String	操作请求ID
switchover_test_records	Array of SwitchoverTestRecord objects	倒换测试记录信息列表
page_info	PageInfo object	分页查询页的信息

表 5-143 SwitchoverTestRecord

参数	参数类型	描述
id	String	倒换测试记录的唯一标识
tenant_id	String	租户ID 最小长度：32 最大长度：32
resource_id	String	倒换测试的资源对象ID
resource_type	String	倒换测试的资源对象类型 缺省值： virtual_interface 最小长度：0 最大长度：128 枚举值： <ul style="list-style-type: none">• virtual_interface
operation	String	shutdown, undo_shutdown表示倒换测试操作类型 最小长度：0 最大长度：128 枚举值： <ul style="list-style-type: none">• shutdown• undo_shutdown
start_time	String	倒换测试操作的开始时间
end_time	String	倒换测试操作的结束时间

参数	参数类型	描述
operate_statuses	String	倒换测试状态记录 STARTING: 初始状态 INPROGRESS: 配置下发中 COMPLETE: 配置下发完成 ERROR: 配置下发失败 枚举值: <ul style="list-style-type: none">● STARTING● INPROGRESS● COMPLETE● ERROR

表 5-144 PageInfo

参数	参数类型	描述
previous_marker	String	上一页的marker, 值为资源的uuid 最小长度: 0 最大长度: 36
current_count	Integer	当前列表中资源数量 最小值: 0 最大值: 2000
next_marker	String	下一页的marker, 值为资源的uuid, 为空时表示最后一页 最小长度: 0 最大长度: 36

请求示例

查询倒换测试记录列表

```
GET https://{dc_endpoint}/v3/de58f033eb664102ba85e4a5db473ca5/dcaas/switchover-test
```

响应示例

状态码: 200

OK

```
{  
  "request_id": "bb154519fb167f99cdb01b7b9995ad14",  
  "switchover_test_records": [  
    {  
      "id": "862d61f1-d9ea-4093-ba0e-2b8d415e3ab3",  
      "tenant_id": "de58f033eb664102ba85e4a5db473ca5",  
      "resource_type": "virtual_interface",  
      "resource_id": "d0b3329c-0063-470c-b1dc-657656b2e540",  
      "operation": "shutdown",  
      "start_time": "2023-10-10T10:41:23.000Z",  
      "end_time": "2023-10-10T10:41:23.000Z",  
    }  
  ]  
}
```

```
"operate_status" : "COMPLETE"  
}]  
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;  
  
import com.huaweicloud.sdk.core.auth.ICredential;  
import com.huaweicloud.sdk.core.auth.BasicCredentials;  
import com.huaweicloud.sdk.core.exception.ConnectionException;  
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;  
import com.huaweicloud.sdk.core.exception.ServiceResponseException;  
import com.huaweicloud.sdk.dc.v3.region.DcRegion;  
import com.huaweicloud.sdk.dc.v3.*;  
import com.huaweicloud.sdk.dc.v3.model.*;  
  
import java.util.List;  
import java.util.ArrayList;  
  
public class ListSwitchoverTestRecordsSolution {  
  
    public static void main(String[] args) {  
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great  
        // security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or  
        // environment variables and decrypted during use to ensure security.  
        // In this example, AK and SK are stored in environment variables for authentication. Before running  
        // this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment  
        String ak = System.getenv("CLOUD_SDK_AK");  
        String sk = System.getenv("CLOUD_SDK_SK");  
  
        ICredential auth = new BasicCredentials()  
            .withAk(ak)  
            .withSk(sk);  
  
        DcClient client = DcClient.newBuilder()  
            .withCredential(auth)  
            .withRegion(DcRegion.valueOf("<YOUR REGION>"))  
            .build();  
        ListSwitchoverTestRecordsRequest request = new ListSwitchoverTestRecordsRequest();  
        request.withLimit("<limit>");  
        request.withMarker("<marker>");  
        request.withFields();  
        request.withSortDir();  
        request.withSortKey("<sort_key>");  
        request.withResourceId();  
        try {  
            ListSwitchoverTestRecordsResponse response = client.listSwitchoverTestRecords(request);  
            System.out.println(response.toString());  
        } catch (ConnectionException e) {  
            e.printStackTrace();  
        } catch (RequestTimeoutException e) {  
            e.printStackTrace();  
        } catch (ServiceResponseException e) {  
            e.printStackTrace();  
            System.out.println(e.getHttpStatusCode());  
            System.out.println(e.getRequestId());  
            System.out.println(e.getErrorCode());  
            System.out.println(e.getErrorMsg());  
        }  
    }  
}
```

Python

```
# coding: utf-8

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdc.v3.region.dc_region import DcRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdc.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    # risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    # variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this
    # example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.getenv("CLOUD_SDK_AK")
    sk = os.getenv("CLOUD_SDK_SK")

    credentials = BasicCredentials(ak, sk) \

    client = DcClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DcRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListSwitchoverTestRecordsRequest()
        request.limit = <limit>
        request.marker = "<marker>"
        request.fields =
        request.sort_dir =
        request.sort_key = "<sort_key>"
        request.resource_id =
        response = client.list_switchover_test_records(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dc "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    // risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    // variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this
    // example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := dc.NewDcClient(
        dc.DcClientBuilder().
```

```
WithRegion(region.ValueOf("<YOUR REGION>")).
WithCredential(auth).
Build()

request := &model.ListSwitchoverTestRecordsRequest{}
limitRequest:= int32(<limit>)
request.Limit = &limitRequest
markerRequest:= "<marker>"
request.Marker = &markerRequest
sortKeyRequest:= "<sort_key>"
request.SortKey = &sortKeyRequest
response, err := client.ListSwitchoverTestRecords(request)
if err == nil {
    fmt.Printf("%+v\n", response)
} else {
    fmt.Println(err)
}
}
```

更多

更多编程语言的SDK代码示例，请参见[API Explorer](#)的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK

错误码

请参见[错误码](#)。

5.4 标签管理

5.4.1 查询项目标签

功能介绍

- 查询租户在指定Project中实例类型的所有资源标签集合。
- 标签管理服务需要能够列出当前租户全部已使用的资源标签集合，为各服务打资源标签和过滤实例时提供标签联想功能。

调用方法

请参见[如何调用API](#)。

URI

GET /v3/{project_id}/{resource_type}/tags

表 5-145 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	租户项目ID
resource_type	是	String	<ul style="list-style-type: none">专线服务资源类型，包括dc-directconnect/dc-vgw/dc-vifdc-directconnect: 专线物理连接dc-vgw: 虚拟网关dc-vif: 虚拟接口 枚举值： <ul style="list-style-type: none">dc-directconnectdc-vgwdc-vif

请求参数

无

响应参数

状态码： 200

表 5-146 响应 Body 参数

参数	参数类型	描述
tags	Array of Tag objects	标签列表。
request_id	String	请求ID

表 5-147 Tag

参数	参数类型	描述
key	String	标签键，最大长度36个unicode字符，格式为大小写字母，数字，中划线“-”，下划线“_”，中文。 最小长度： 0 最大长度： 36

参数	参数类型	描述
value	String	标签值，最大长度43个unicode字符，格式为大写字母，数字，中划线“-”，下划线“_”，点“.”，中文。 最小长度： 0 最大长度： 43

请求示例

```
GET https://{dc_endpoint}/v3/ed28c294165741faeccab26913122a1/dc-directconnect/tags
```

响应示例

状态码： 200

OK

```
{
  "tags": [ {
    "key": "department",
    "value": "finance"
  } ]
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.dc.v3.region.DcRegion;
import com.huaweicloud.sdk.dc.v3.*;
import com.huaweicloud.sdk.dc.v3.model.*;

public class ListProjectTagsSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
        // security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
        // environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running
        // this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
            .withAk(ak)
            .withSk(sk);

        DcClient client = DcClient.newBuilder()
            .withCredential(auth)
            .withRegion(DcRegion.valueOf("<YOUR REGION>"))
```

```
        .build();
ListProjectTagsRequest request = new ListProjectTagsRequest();
try {
    ListProjectTagsResponse response = client.listProjectTags(request);
    System.out.println(response.toString());
} catch (ConnectionException e) {
    e.printStackTrace();
} catch (RequestTimeoutException e) {
    e.printStackTrace();
} catch (ServiceResponseException e) {
    e.printStackTrace();
    System.out.println(e.getStatusCode());
    System.out.println(e.getRequestId());
    System.out.println(e.getErrorCode());
    System.out.println(e.getErrorMsg());
}
}
```

Python

```
# coding: utf-8

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdc.v3.region.dc_region import DcRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdc.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    # risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    # variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this
    # example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.getenv("CLOUD_SDK_AK")
    sk = os.getenv("CLOUD_SDK_SK")

    credentials = BasicCredentials(ak, sk) \

    client = DcClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DcRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListProjectTagsRequest()
        response = client.list_project_tags(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dc "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    // risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
```

```
variables and decrypted during use to ensure security.
// In this example, AK and SK are stored in environment variables for authentication. Before running this
example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
ak := os.Getenv("CLOUD_SDK_AK")
sk := os.Getenv("CLOUD_SDK_SK")

auth := basic.NewCredentialsBuilder().
    WithAk(ak).
    WithSk(sk).
    Build()

client := dc.NewDcClient(
    dc.DcClientBuilder().
        WithRegion(region.ValueOf("<YOUR REGION>")).
        WithCredential(auth).
        Build())

request := &model.ListProjectTagsRequest{}
response, err := client.ListProjectTags(request)
if err == nil {
    fmt.Printf("%+v\n", response)
} else {
    fmt.Println(err)
}
}
```

更多

更多编程语言的SDK代码示例，请参见[API Explorer](#)的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK

错误码

请参见[错误码](#)。

5.4.2 查询资源标签

功能介绍

查询资源标签

调用方法

请参见[如何调用API](#)。

URI

GET /v3/{project_id}/{resource_type}/{resource_id}/tags

表 5-148 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	租户项目ID
resource_type	是	String	<ul style="list-style-type: none">专线服务资源类型，包括dc-directconnect/dc-vgw/dc-vifdc-directconnect: 专线物理连接dc-vgw: 虚拟网关dc-vif: 虚拟接口 枚举值： <ul style="list-style-type: none">dc-directconnectdc-vgwdc-vif
resource_id	是	String	资源实例ID

请求参数

无

响应参数

状态码： 200

表 5-149 响应 Body 参数

参数	参数类型	描述
tags	Array of Tag objects	标签列表
sys_tags	Array of Tag objects	标签列表，没有标签默认为空数组。
request_id	String	请求ID

表 5-150 Tag

参数	参数类型	描述
key	String	标签键，最大长度36个unicode字符，格式为大小写字母，数字，中划线“-”，下划线“_”，中文。 最小长度： 0 最大长度： 36

参数	参数类型	描述
value	String	标签值，最大长度43个unicode字符，格式为大写字母，数字，中划线“-”，下划线“_”，点“.”，中文。 最小长度： 0 最大长度： 43

请求示例

查询资源标签

```
https://{endpoint}/v3/cfa563efb77d4b6d9960781d82530fd8/dc-vgw/resource-instances/action
```

响应示例

状态码：**200**

OK

```
{
  "request_id" : "80ef5f21-b81a-4546-b23d-84272507d330",
  "tags" : [ {
    "key" : "key2",
    "value" : "value2"
  }, {
    "key" : "key1",
    "value" : "value1"
  }, {
    "key" : "key3",
    "value" : "value3"
  } ]
}
```

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.dc.v3.region.DcRegion;
import com.huaweicloud.sdk.dc.v3.*;
import com.huaweicloud.sdk.dc.v3.model.*;

public class ShowResourceTagSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
        // security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
        // environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running
        // this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    }
}
```

```
String ak = System.getenv("CLOUD_SDK_AK");
String sk = System.getenv("CLOUD_SDK_SK");

ICredential auth = new BasicCredentials()
    .withAk(ak)
    .withSk(sk);

DcClient client = DcClient.newBuilder()
    .withCredential(auth)
    .withRegion(DcRegion.valueOf("<YOUR REGION>"))
    .build();
ShowResourceTagRequest request = new ShowResourceTagRequest();
try {
    ShowResourceTagResponse response = client.showResourceTag(request);
    System.out.println(response.toString());
} catch (ConnectionException e) {
    e.printStackTrace();
} catch (RequestTimeoutException e) {
    e.printStackTrace();
} catch (ServiceResponseException e) {
    e.printStackTrace();
    System.out.println(e.getStatusCode());
    System.out.println(e.getRequestId());
    System.out.println(e.getErrorCode());
    System.out.println(e.getErrorMsg());
}
}
```

Python

```
# coding: utf-8

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdc.v3.region.dc_region import DcRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdc.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    # risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    # variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this
    # example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.getenv("CLOUD_SDK_AK")
    sk = os.getenv("CLOUD_SDK_SK")

    credentials = BasicCredentials(ak, sk) \

    client = DcClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DcRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ShowResourceTagRequest()
        response = client.show_resource_tag(request)
        print(response)
    except exceptions.ClientRequestException as e:
        print(e.status_code)
        print(e.request_id)
        print(e.error_code)
        print(e.error_msg)
```

Go

```
package main

import (
```

```
"fmt"
"github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
dc "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3"
"github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/model"
region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this
    example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := dc.NewDcClient(
        dc.DcClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowResourceTagRequest{}
    response, err := client.ShowResourceTag(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见[API Explorer](#)的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK

错误码

请参见[错误码](#)。

5.4.3 添加资源标签

功能介绍

- 一个资源上最多有10个标签。
- 此接口为幂等接口：
- 创建时，如果创建的标签已经存在（key相同），则覆盖。

调用方法

请参见[如何调用API](#)。

URI

POST /v3/{project_id}/{resource_type}/{resource_id}/tags

表 5-151 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	租户项目ID
resource_id	是	String	资源实例ID
resource_type	是	String	<ul style="list-style-type: none">专线服务资源类型，包括dc-directconnect/dc-vgw/dc-vifdc-directconnect: 专线物理连接dc-vgw: 虚拟网关dc-vif: 虚拟接口 枚举值： <ul style="list-style-type: none">dc-directconnectdc-vgwdc-vif

请求参数

表 5-152 请求 Body 参数

参数	是否必选	参数类型	描述
tag	是	Tag object	资源标签。

表 5-153 Tag

参数	是否必选	参数类型	描述
key	是	String	标签键，最大长度36个unicode字符，格式为大小写字母，数字，中划线“-”，下划线“_”，中文。 最小长度： 0 最大长度： 36

参数	是否必选	参数类型	描述
value	否	String	标签值，最大长度43个unicode字符，格式为大小写字母，数字，中划线“-”，下划线“_”，点“.”，中文。 最小长度： 0 最大长度： 43

响应参数

无

请求示例

添加资源标签

```
POST https://{endpoint}/v3/cfa563efb77d4b6d9960781d82530fd8/dc-vgw/3320166e-b937-40cc-a35c-02cd3f2b3ee2/tags
```

```
{
  "tag": {
    "key": "key1",
    "value": "value1"
  }
}
```

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

添加资源标签

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.dc.v3.region.DcRegion;
import com.huaweicloud.sdk.dc.v3.*;
import com.huaweicloud.sdk.dc.v3.model.*;

public class CreateResourceTagSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
        // security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
        // environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running
```

this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment

```
String ak = System.getenv("CLOUD_SDK_AK");
String sk = System.getenv("CLOUD_SDK_SK");

ICredential auth = new BasicCredentials()
    .withAk(ak)
    .withSk(sk);

DcClient client = DcClient.newBuilder()
    .withCredential(auth)
    .withRegion(DcRegion.valueOf("<YOUR REGION>"))
    .build();
CreateResourceTagRequest request = new CreateResourceTagRequest();
CreateResourceTagRequestBody body = new CreateResourceTagRequestBody();
Tag tagbody = new Tag();
tagbody.withKey("key1")
    .withValue("value1");
body.withTag(tagbody);
request.withBody(body);
try {
    CreateResourceTagResponse response = client.createResourceTag(request);
    System.out.println(response.toString());
} catch (ConnectionException e) {
    e.printStackTrace();
} catch (RequestTimeoutException e) {
    e.printStackTrace();
} catch (ServiceResponseException e) {
    e.printStackTrace();
    System.out.println(e.getHttpStatusCode());
    System.out.println(e.getRequestId());
    System.out.println(e.getErrorCode());
    System.out.println(e.getErrorMsg());
}
}
```

Python

添加资源标签

```
# coding: utf-8

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdc.v3.region.dc_region import DcRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdc.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    # risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    # variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this
    # example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.getenv("CLOUD_SDK_AK")
    sk = os.getenv("CLOUD_SDK_SK")

    credentials = BasicCredentials(ak, sk) \

    client = DcClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DcRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = CreateResourceTagRequest()
        tagbody = Tag(
            key="key1",
            value="value1"
        )
```

```
request.body = CreateResourceTagRequestBody(  
    tag=tagbody  
)  
response = client.create_resource_tag(request)  
print(response)  
except exceptions.ClientRequestException as e:  
    print(e.status_code)  
    print(e.request_id)  
    print(e.error_code)  
    print(e.error_msg)
```

Go

添加资源标签

```
package main  
  
import (  
    "fmt"  
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"  
    dc "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3"  
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/model"  
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/region"  
)  
  
func main() {  
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security  
    // risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment  
    // variables and decrypted during use to ensure security.  
    // In this example, AK and SK are stored in environment variables for authentication. Before running this  
    // example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment  
    ak := os.Getenv("CLOUD_SDK_AK")  
    sk := os.Getenv("CLOUD_SDK_SK")  
  
    auth := basic.NewCredentialsBuilder().  
        WithAk(ak).  
        WithSk(sk).  
        Build()  
  
    client := dc.NewDcClient(  
        dc.DcClientBuilder().  
            WithRegion(region.ValueOf("<YOUR REGION>")).  
            WithCredential(auth).  
            Build())  
  
    request := &model.CreateResourceTagRequest{  
        valueTag:= "value1"  
        tagbody := &model.Tag{  
            Key: "key1",  
            Value: &valueTag,  
        }  
        request.Body = &model.CreateResourceTagRequestBody{  
            Tag: tagbody,  
        }  
        response, err := client.CreateResourceTag(request)  
        if err == nil {  
            fmt.Printf("%+v\n", response)  
        } else {  
            fmt.Println(err)  
        }  
    }  
}
```

更多

更多编程语言的SDK代码示例，请参见[API Explorer](#)的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
204	No Content

错误码

请参见[错误码](#)。

5.4.4 批量添加删除资源标签

功能介绍

- 为指定实例批量添加或删除标签
- 标签管理服务需要使用该接口批量管理实例的标签。
- 一个资源上最多有10个标签。

调用方法

请参见[如何调用API](#)。

URI

POST /v3/{project_id}/{resource_type}/{resource_id}/tags/action

表 5-154 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	租户项目ID
resource_id	是	String	资源实例ID
resource_type	是	String	<ul style="list-style-type: none">• 专线服务资源类型，包括dc-directconnect/dc-vgw/dc-vif• dc-directconnect: 专线物理连接• dc-vgw: 虚拟网关• dc-vif: 虚拟接口 枚举值： <ul style="list-style-type: none">• dc-directconnect• dc-vgw• dc-vif

请求参数

表 5-155 请求 Body 参数

参数	是否必选	参数类型	描述
action	是	String	功能说明：操作标识。取值范围：create（创建）delete（删除） 枚举值： <ul style="list-style-type: none">• create• delete
tags	否	Array of Tag objects	标签列表。
sys_tags	否	Array of Tag objects	标签列表。

表 5-156 Tag

参数	是否必选	参数类型	描述
key	是	String	标签键，最大长度36个unicode字符，格式为大小写字母，数字，中划线“-”，下划线“_”，中文。 最小长度： 0 最大长度： 36
value	否	String	标签值，最大长度43个unicode字符，格式为大小写字母，数字，中划线“-”，下划线“_”，点“.”，中文。 最小长度： 0 最大长度： 43

响应参数

无

请求示例

- 批量添加资源标签。
POST https://{endpoint}/v3/cfa563efb77d4b6d9960781d82530fd8/dc-vgw/3320166e-b937-40cc-a35c-02cd3f2b3ee2/tags/action

```
{
  "action": "create",
  "tags": [{
```

```
"key" : "key1",  
"value" : "value1"  
}, {  
"key" : "key2",  
"value" : "value2"  
}]  
}
```

- 批量删除资源标签。

POST https://{endpoint}/v3/cfa563efb77d4b6d9960781d82530fd8/dc-vgw/3320166e-b937-40cc-a35c-02cd3f2b3ee2/tags/action

```
{  
"action" : "delete",  
"tags" : [ {  
"key" : "key1"  
}, {  
"key" : "key2",  
"value" : "value3"  
}]  
}
```

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

- 批量添加资源标签。

```
package com.huaweicloud.sdk.test;  
  
import com.huaweicloud.sdk.core.auth.ICredential;  
import com.huaweicloud.sdk.core.auth.BasicCredentials;  
import com.huaweicloud.sdk.core.exception.ConnectionException;  
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;  
import com.huaweicloud.sdk.core.exception.ServiceResponseException;  
import com.huaweicloud.sdk.dc.v3.region.DcRegion;  
import com.huaweicloud.sdk.dc.v3.*;  
import com.huaweicloud.sdk.dc.v3.model.*;  
  
import java.util.List;  
import java.util.ArrayList;  
  
public class BatchCreateResourceTagsSolution {  
  
    public static void main(String[] args) {  
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great  
        // security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or  
        // environment variables and decrypted during use to ensure security.  
        // In this example, AK and SK are stored in environment variables for authentication. Before  
        // running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local  
        // environment  
        String ak = System.getenv("CLOUD_SDK_AK");  
        String sk = System.getenv("CLOUD_SDK_SK");  
  
        ICredential auth = new BasicCredentials()  
            .withAk(ak)  
            .withSk(sk);  
  
        DcClient client = DcClient.newBuilder()  
            .withCredential(auth)  
            .withRegion(DcRegion.valueOf("<YOUR REGION>"))  
            .build();
```

```
BatchCreateResourceTagsRequest request = new BatchCreateResourceTagsRequest();
BatchOperateResourceTagsRequestBody body = new BatchOperateResourceTagsRequestBody();
List<Tag> listbodyTags = new ArrayList<>();
listbodyTags.add(
    new Tag()
        .withKey("key1")
        .withValue("value1")
);
listbodyTags.add(
    new Tag()
        .withKey("key2")
        .withValue("value2")
);
body.withTags(listbodyTags);
body.withAction(BatchOperateResourceTagsRequestBody.ActionEnum.fromValue("create"));
request.withBody(body);
try {
    BatchCreateResourceTagsResponse response = client.batchCreateResourceTags(request);
    System.out.println(response.toString());
} catch (ConnectionException e) {
    e.printStackTrace();
} catch (RequestTimeoutException e) {
    e.printStackTrace();
} catch (ServiceResponseException e) {
    e.printStackTrace();
    System.out.println(e.getHttpStatusCode());
    System.out.println(e.getRequestId());
    System.out.println(e.getErrorCode());
    System.out.println(e.getErrorMsg());
}
}
```

- **批量删除资源标签。**

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.dc.v3.region.DcRegion;
import com.huaweicloud.sdk.dc.v3.*;
import com.huaweicloud.sdk.dc.v3.model.*;

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

public class BatchCreateResourceTagsSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
        // security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
        // environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before
        // running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local
        // environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
            .withAk(ak)
            .withSk(sk);

        DcClient client = DcClient.newBuilder()
            .withCredential(auth)
            .withRegion(DcRegion.valueOf("<YOUR REGION>"))
            .build();
        BatchCreateResourceTagsRequest request = new BatchCreateResourceTagsRequest();
        BatchOperateResourceTagsRequestBody body = new BatchOperateResourceTagsRequestBody();
```



```
List<Tag> listbodyTags = new ArrayList<>();
listbodyTags.add(
    new Tag()
        .withKey("key1")
);
listbodyTags.add(
    new Tag()
        .withKey("key2")
        .withValue("value3")
);
body.withTags(listbodyTags);
body.withAction(BatchOperateResourceTagsRequestBody.ActionEnum.fromValue("delete"));
request.withBody(body);
try {
    BatchCreateResourceTagsResponse response = client.batchCreateResourceTags(request);
    System.out.println(response.toString());
} catch (ConnectionException e) {
    e.printStackTrace();
} catch (RequestTimeoutException e) {
    e.printStackTrace();
} catch (ServiceResponseException e) {
    e.printStackTrace();
    System.out.println(e.getHttpStatusCode());
    System.out.println(e.getRequestId());
    System.out.println(e.getErrorCode());
    System.out.println(e.getErrorMsg());
}
}
```

Python

- 批量添加资源标签。

```
# coding: utf-8

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdc.v3.region.dc_region import DcRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdc.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
    # security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
    # environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before
    # running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local
    # environment
    ak = os.getenv("CLOUD_SDK_AK")
    sk = os.getenv("CLOUD_SDK_SK")

    credentials = BasicCredentials(ak, sk) \

    client = DcClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DcRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = BatchCreateResourceTagsRequest()
        listTagsbody = [
            Tag(
                key="key1",
                value="value1"
            ),
            Tag(
                key="key2",
                value="value2"
            )
        ]
```

```
request.body = BatchOperateResourceTagsRequestBody(  
    tags=listTagsbody,  
    action="create"  
)  
response = client.batch_create_resource_tags(request)  
print(response)  
except exceptions.ClientRequestException as e:  
    print(e.status_code)  
    print(e.request_id)  
    print(e.error_code)  
    print(e.error_msg)
```

- 批量删除资源标签。

```
# coding: utf-8
```

```
from huaweicloudsdkcore.auth.credentials import BasicCredentials  
from huaweicloudsdkdc.v3.region.dc_region import DcRegion  
from huaweicloudsdkcore.exceptions import exceptions  
from huaweicloudsdkdc.v3 import *
```

```
if __name__ == "__main__":
```

```
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great  
    security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or  
    environment variables and decrypted during use to ensure security.
```

```
    # In this example, AK and SK are stored in environment variables for authentication. Before  
    running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local  
    environment
```

```
    ak = os.getenv("CLOUD_SDK_AK")  
    sk = os.getenv("CLOUD_SDK_SK")
```

```
    credentials = BasicCredentials(ak, sk) \
```

```
    client = DcClient.new_builder() \  
        .with_credentials(credentials) \  
        .with_region(DcRegion.value_of("<YOUR REGION>")) \  
        .build()
```

```
try:
```

```
    request = BatchCreateResourceTagsRequest()
```

```
    listTagsbody = [  
        Tag(  
            key="key1"        ),  
        Tag(  
            key="key2",  
            value="value3"        )  
    ]
```

```
    ]  
    request.body = BatchOperateResourceTagsRequestBody(  
        tags=listTagsbody,  
        action="delete"  
    )  
    response = client.batch_create_resource_tags(request)  
    print(response)  
except exceptions.ClientRequestException as e:  
    print(e.status_code)  
    print(e.request_id)  
    print(e.error_code)  
    print(e.error_msg)
```

```
    print(e.status_code)  
    print(e.request_id)  
    print(e.error_code)  
    print(e.error_msg)
```

```
    print(e.status_code)  
    print(e.request_id)  
    print(e.error_code)  
    print(e.error_msg)
```

```
except exceptions.ClientRequestException as e:
```

```
    print(e.status_code)  
    print(e.request_id)  
    print(e.error_code)  
    print(e.error_msg)
```

Go

- 批量添加资源标签。

```
package main
```

```
import (  
    "fmt"    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"    dc "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3")
```

```
func main() {  
    // ...  
}
```

```
"github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/model"
region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
    // security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
    // environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before
    // running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local
    // environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := dc.NewDcClient(
        dc.DcClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.BatchCreateResourceTagsRequest{
        valueTags:= "value1"
        valueTags1:= "value2"
        var listTagsbody = []model.Tag{
            {
                Key: "key1",
                Value: &valueTags,
            },
            {
                Key: "key2",
                Value: &valueTags1,
            },
        }
        request.Body = &model.BatchOperateResourceTagsRequestBody{
            Tags: &listTagsbody,
            Action: model.GetBatchOperateResourceTagsRequestBodyActionEnum().CREATE,
        }
        response, err := client.BatchCreateResourceTags(request)
        if err == nil {
            fmt.Printf("%+v\n", response)
        } else {
            fmt.Println(err)
        }
    }
}
```

- 批量删除资源标签。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dc "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
    // security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
    // environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before
    // running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local
    // environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")
}
```

```
auth := basic.NewCredentialsBuilder().
    WithAk(ak).
    WithSk(sk).
    Build()

client := dc.NewDcClient(
    dc.DcClientBuilder().
        WithRegion(region.ValueOf("<YOUR REGION>")).
        WithCredential(auth).
        Build())

request := &model.BatchCreateResourceTagsRequest{
    valueTags:= "value3"
    var listTagsbody = []model.Tag{
        {
            Key: "key1",
        },
        {
            Key: "key2",
            Value: &valueTags,
        },
    }
    request.Body = &model.BatchOperateResourceTagsRequestBody{
        Tags: &listTagsbody,
        Action: model.GetBatchOperateResourceTagsRequestBodyActionEnum().DELETE,
    }
    response, err := client.BatchCreateResourceTags(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见[API Explorer](#)的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
204	No Content

错误码

请参见[错误码](#)。

5.4.5 删除资源标签

功能介绍

删除时,不对标签字符集做校验，调用接口前必须要做encodeURIComponent，服务端需要对接口uri做decodeURI。删除的key不存在报404，Key不能为空或者空字符串。

调用方法

请参见[如何调用API](#)。

URI

DELETE /v3/{project_id}/{resource_type}/{resource_id}/tags/{key}

表 5-157 路径参数

参数	是否必选	参数类型	描述
key	是	String	标签key。
project_id	是	String	租户项目ID
resource_id	是	String	资源实例ID
resource_type	是	String	<ul style="list-style-type: none">专线服务资源类型，包括dc-directconnect/dc-vgw/dc-vifdc-directconnect: 专线物理连接dc-vgw: 虚拟网关dc-vif: 虚拟接口 枚举值： <ul style="list-style-type: none">dc-directconnectdc-vgwdc-vif

请求参数

无

响应参数

无

请求示例

删除资源标签

```
DELETE https://{dc-endpoint}/v3/cfa563efb77d4b6d9960781d82530fd8/instance/3320166e-b937-40cc-a35c-02cd3f2b3ee2/tags/key1
```

响应示例

无

SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.dc.v3.region.DcRegion;
import com.huaweicloud.sdk.dc.v3.*;
import com.huaweicloud.sdk.dc.v3.model.*;

public class DeleteResourceTagSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
        // security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
        // environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running
        // this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
            .withAk(ak)
            .withSk(sk);

        DcClient client = DcClient.newBuilder()
            .withCredential(auth)
            .withRegion(DcRegion.valueOf("<YOUR REGION>"))
            .build();
        DeleteResourceTagRequest request = new DeleteResourceTagRequest();
        try {
            DeleteResourceTagResponse response = client.deleteResourceTag(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdc.v3.region.dc_region import DcRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdc.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    # risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    # variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before running this
    # example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak = os.getenv("CLOUD_SDK_AK")
    sk = os.getenv("CLOUD_SDK_SK")
```

```
credentials = BasicCredentials(ak, sk) \

client = DcClient.new_builder() \
    .with_credentials(credentials) \
    .with_region(DcRegion.value_of("<YOUR REGION>")) \
    .build()

try:
    request = DeleteResourceTagRequest()
    response = client.delete_resource_tag(request)
    print(response)
except exceptions.ClientRequestException as e:
    print(e.status_code)
    print(e.request_id)
    print(e.error_code)
    print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dc "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security
    // risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment
    // variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this
    // example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := dc.NewDcClient(
        dc.DcClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.DeleteResourceTagRequest{}
    response, err := client.DeleteResourceTag(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见[API Explorer](#)的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
204	No Content

错误码

请参见[错误码](#)。

5.4.6 通过标签查询资源实例

功能介绍

通过标签查询资源实例

调用方法

请参见[如何调用API](#)。

URI

POST /v3/{project_id}/{resource_type}/resource-instances/action

表 5-158 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	租户项目ID
resource_type	是	String	<ul style="list-style-type: none">专线服务资源类型，包括dc-directconnect/dc-vgw/dc-vifdc-directconnect: 专线物理连接dc-vgw: 虚拟网关dc-vif: 虚拟接口 枚举值: <ul style="list-style-type: none">dc-directconnectdc-vgwdc-vif

请求参数

表 5-159 请求 Body 参数

参数	是否必选	参数类型	描述
offset	否	String	索引位置，从offset指定的下一条数据开始查询。查询第一页数据时，不需要传入此参数，查询后续页码数据时，将查询前一页数据时响应体中的值带入此参数（action为count时无此参数）如果action为filter默认为0，必须为数字，不能为负数
limit	否	String	查询记录数（action为count时无此参数）如果action为filter默认为1000，limit最多为1000,不能为负数，最小值为1
action	是	String	操作标识（仅限于filter，count）：filter（过滤），count(查询总条数) 如果是filter就是分页查询，如果是count只需按照条件将总条数返回即可。禁止返回其他字段。 枚举值： <ul style="list-style-type: none">• filter• count
matches	否	Array of Match objects	搜索字段,key为要匹配的字段，如resource_name等。value为匹配的值。此字段为固定字典值。根据不同的字段确认是否需要模糊匹配，如resource_name默认为模糊搜索（不区分大小写），如果value为空字符串精确匹配。resource_id为精确匹配。第一期只做resource_name，后续在扩展。
not_tags	否	Array of Tags objects	不包含标签，最多包含10个key，每个key下面的value最多10个，结构体不能缺失，key不能为空或者空字符串。Key不能重复，同一个key中values不能重复。返回不包含标签的资源列表，key之间是与的关系，key-value结构中value是或的关系。无过滤条件时返回全量数据。

参数	是否必选	参数类型	描述
tags	否	Array of Tags objects	包含标签，最多包含10个key，每个key下面的value最多10个，结构体不能缺失，key不能为空或者空字符串。Key不能重复，同一个key中values不能重复。返回包含所有标签的资源列表，key之间是与的关系，key-value结构中value是或的关系。无tag过滤条件时返回全量数据。
tags_any	否	Array of Tags objects	包含任意标签，最多包含10个key，每个key下面的value最多10个，结构体不能缺失，key不能为空或者空字符串。Key不能重复，同一个key中values不能重复。返回包含标签的资源列表，key之间是或的关系，key-value结构中value是或的关系。无过滤条件时返回全量数据。
not_tags_any	否	Array of Tags objects	不包含任意标签，最多包含10个key，每个key下面的value最多10个，结构体不能缺失，key不能为空或者空字符串。Key不能重复，同一个key中values不能重复。返回不包含标签的资源列表，key之间是或的关系，key-value结构中value是或的关系。无过滤条件时返回全量数据。
sys_tags	否	Array of Tags objects	仅op_service权限可以使用此字段做资源实例过滤条件。目前TMS调用时只包含一个tag结构体。key： _sys_enterprise_project_id， value：企业项目id列表。目前TMS调用时，key下面只包含一个value。0表示默认企业项目。sys_tags和租户标签过滤条件（without_any_tag、tags、tags_any、not_tags、not_tags_any）不能同时使用。

表 5-160 Match

参数	是否必选	参数类型	描述
key	是	String	键。取值范围为包括dc-directconnect/dc-vgw/dc-vif

参数	是否必选	参数类型	描述
value	是	String	值。每个值最大长度255个unicode字符。

表 5-161 Tags

参数	是否必选	参数类型	描述
key	是	String	键。最大长度127个unicode字符。key不能为空。
values	是	Array of strings	值列表。每个值最大长度255个unicode字符。

响应参数

状态码： 200

表 5-162 响应 Body 参数

参数	参数类型	描述
resources	Array of Resource objects	资源列表。
total_count	Integer	总记录数。
request_id	String	请求ID

表 5-163 Resource

参数	参数类型	描述
resource_detail	Object	资源详情。资源对象，用于扩展。默认为空。
resource_id	String	资源的ID。
resource_name	String	资源名称，资源没有名称时默认为空字符串。
tags	Array of Tag objects	标签列表，没有标签默认为空数组。
sys_tags	Array of Tag objects	标签列表，没有标签默认为空数组。

表 5-164 Tag

参数	参数类型	描述
key	String	标签键，最大长度36个unicode字符，格式为大小写字母，数字，中划线“-”，下划线“_”，中文。 最小长度： 0 最大长度： 36
value	String	标签值，最大长度43个unicode字符，格式为大小写字母，数字，中划线“-”，下划线“_”，点“.”，中文。 最小长度： 0 最大长度： 43

请求示例

- action为filter时，根据标签查询资源。

<https://{endpoint}/v3/cfa563efb77d4b6d9960781d82530fd8/dc-vgw/resource-instances/action>

```
{
  "offset": "10",
  "limit": "10",
  "action": "filter",
  "matches": [ {
    "key": "resource_name",
    "value": "resource1"
  } ],
  "not_tags": [ {
    "key": "key1",
    "values": [ "*"value1", "value2" ]
  } ],
  "tags": [ {
    "key": "key1",
    "values": [ "*"value1", "value2" ]
  } ],
  "tags_any": [ {
    "key": "key1",
    "values": [ "value1", "value2" ]
  } ],
  "not_tags_any": [ {
    "key": "key1",
    "values": [ "value1", "value2" ]
  } ]
}
```

- action为count时，查询资源总数。

https://{endpoint}/v3/cfa563efb77d4b6d9960781d82530fd8/dc-vgw/resource_instances/action

```
{
  "action": "count",
  "not_tags": [ {
    "key": "key1",
    "values": [ "value1", "*"value2" ]
  } ],
  "tags": [ {
    "key": "key1",
    "values": [ "value1", "value2" ]
  } ], {
    "key": "key2",
    "values": [ "value1", "value2" ]
  } ]
}
```

```
    }],  
    "tags_any": [ {  
      "key": "key1",  
      "values": [ "value1", "value2" ]  
    }],  
    "not_tags_any": [ {  
      "key": "key1",  
      "values": [ "value1", "value2" ]  
    }],  
    "sys_tags": [ {  
      "key": "_sys_enterprise_project_id",  
      "values": [ "5aa119a8-d25b-45a7-8d1b-88e127885635" ]  
    }],  
    "matches": [ {  
      "key": "resource_name",  
      "value": "resource1"  
    }]  
  }  
}
```

响应示例

状态码： 200

OK

```
{  
  "resources": [],  
  "total_count": 0,  
  "request_id": "9a4f4dfc4fb2fc101e65bba07d908535"  
}
```

SDK 代码示例

SDK代码示例如下。

Java

- action为filter时，根据标签查询资源。

```
package com.huaweicloud.sdk.test;  
  
import com.huaweicloud.sdk.core.auth.ICredential;  
import com.huaweicloud.sdk.core.auth.BasicCredentials;  
import com.huaweicloud.sdk.core.exception.ConnectionException;  
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;  
import com.huaweicloud.sdk.core.exception.ServiceResponseException;  
import com.huaweicloud.sdk.dc.v3.region.DcRegion;  
import com.huaweicloud.sdk.dc.v3.*;  
import com.huaweicloud.sdk.dc.v3.model.*;  
  
import java.util.List;  
import java.util.ArrayList;  
  
public class ListTagResourceInstancesSolution {  
  
    public static void main(String[] args) {  
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great  
        // security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or  
        // environment variables and decrypted during use to ensure security.  
        // In this example, AK and SK are stored in environment variables for authentication. Before  
        // running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local  
        // environment  
        String ak = System.getenv("CLOUD_SDK_AK");  
        String sk = System.getenv("CLOUD_SDK_SK");  
  
        ICredential auth = new BasicCredentials()  
            .withAk(ak)
```

```
.withSk(sk);

DcClient client = DcClient.newBuilder()
    .withCredential(auth)
    .withRegion(DcRegion.valueOf("<YOUR REGION>"))
    .build();
ListTagResourceInstancesRequest request = new ListTagResourceInstancesRequest();
ListTagResourceInstancesRequestBody body = new ListTagResourceInstancesRequestBody();
List<String> listNotTagsAnyValues = new ArrayList<>();
listNotTagsAnyValues.add("value1");
listNotTagsAnyValues.add("value2");
List<Tags> listbodyNotTagsAny = new ArrayList<>();
listbodyNotTagsAny.add(
    new Tags()
        .withKey("key1")
        .withValues(listNotTagsAnyValues)
);
List<String> listTagsAnyValues = new ArrayList<>();
listTagsAnyValues.add("value1");
listTagsAnyValues.add("value2");
List<Tags> listbodyTagsAny = new ArrayList<>();
listbodyTagsAny.add(
    new Tags()
        .withKey("key1")
        .withValues(listTagsAnyValues)
);
List<String> listTagsValues = new ArrayList<>();
listTagsValues.add("*value1");
listTagsValues.add("value2");
List<Tags> listbodyTags = new ArrayList<>();
listbodyTags.add(
    new Tags()
        .withKey("key1")
        .withValues(listTagsValues)
);
List<String> listNotTagsValues = new ArrayList<>();
listNotTagsValues.add("*value1");
listNotTagsValues.add("value2");
List<Tags> listbodyNotTags = new ArrayList<>();
listbodyNotTags.add(
    new Tags()
        .withKey("key1")
        .withValues(listNotTagsValues)
);
List<Match> listbodyMatches = new ArrayList<>();
listbodyMatches.add(
    new Match()
        .withKey("resource_name")
        .withValue("resource1")
);
body.withNotTagsAny(listbodyNotTagsAny);
body.withTagsAny(listbodyTagsAny);
body.withTags(listbodyTags);
body.withNotTags(listbodyNotTags);
body.withMatches(listbodyMatches);
body.withAction(ListTagResourceInstancesRequestBody.ActionEnum.fromValue("filter"));
body.withLimit("10");
body.withOffset("10");
request.withBody(body);
try {
    ListTagResourceInstancesResponse response = client.listTagResourceInstances(request);
    System.out.println(response.toString());
} catch (ConnectionException e) {
    e.printStackTrace();
} catch (RequestTimeoutException e) {
    e.printStackTrace();
} catch (ServiceResponseException e) {
    e.printStackTrace();
    System.out.println(e.getHttpStatusCode());
}
```

```
        System.out.println(e.getRequestId());
        System.out.println(e.getErrorCode());
        System.out.println(e.getErrorMsg());
    }
}
}
```

- **action为count时，查询资源总数。**

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.dc.v3.region.DcRegion;
import com.huaweicloud.sdk.dc.v3.*;
import com.huaweicloud.sdk.dc.v3.model.*;

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

public class ListTagResourceInstancesSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
        // security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
        // environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before
        // running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local
        // environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
            .withAk(ak)
            .withSk(sk);

        DcClient client = DcClient.newBuilder()
            .withCredential(auth)
            .withRegion(DcRegion.valueOf("<YOUR REGION>"))
            .build();
        ListTagResourceInstancesRequest request = new ListTagResourceInstancesRequest();
        ListTagResourceInstancesRequestBody body = new ListTagResourceInstancesRequestBody();
        List<String> listSysTagsValues = new ArrayList<>();
        listSysTagsValues.add("5aa119a8-d25b-45a7-8d1b-88e127885635");
        List<Tags> listbodySysTags = new ArrayList<>();
        listbodySysTags.add(
            new Tags()
                .withKey("_sys_enterprise_project_id")
                .withValues(listSysTagsValues)
        );
        List<String> listNotTagsAnyValues = new ArrayList<>();
        listNotTagsAnyValues.add("value1");
        listNotTagsAnyValues.add("value2");
        List<Tags> listbodyNotTagsAny = new ArrayList<>();
        listbodyNotTagsAny.add(
            new Tags()
                .withKey("key1")
                .withValues(listNotTagsAnyValues)
        );
        List<String> listTagsAnyValues = new ArrayList<>();
        listTagsAnyValues.add("value1");
        listTagsAnyValues.add("value2");
        List<Tags> listbodyTagsAny = new ArrayList<>();
        listbodyTagsAny.add(
            new Tags()
                .withKey("key1")
                .withValues(listTagsAnyValues)
        );
    }
}
```

```
List<String> listTagsValues = new ArrayList<>();
listTagsValues.add("value1");
listTagsValues.add("value2");
List<String> listTagsValues1 = new ArrayList<>();
listTagsValues1.add("value1");
listTagsValues1.add("value2");
List<Tags> listbodyTags = new ArrayList<>();
listbodyTags.add(
    new Tags()
        .withKey("key1")
        .withValues(listTagsValues1)
);
listbodyTags.add(
    new Tags()
        .withKey("key2")
        .withValues(listTagsValues)
);
List<String> listNotTagsValues = new ArrayList<>();
listNotTagsValues.add("value1");
listNotTagsValues.add("*value2");
List<Tags> listbodyNotTags = new ArrayList<>();
listbodyNotTags.add(
    new Tags()
        .withKey("key1")
        .withValues(listNotTagsValues)
);
List<Match> listbodyMatches = new ArrayList<>();
listbodyMatches.add(
    new Match()
        .withKey("resource_name")
        .withValue("resource1")
);
body.withSysTags(listbodySysTags);
body.withNotTagsAny(listbodyNotTagsAny);
body.withTagsAny(listbodyTagsAny);
body.withTags(listbodyTags);
body.withNotTags(listbodyNotTags);
body.withMatches(listbodyMatches);
body.withAction(ListTagResourceInstancesRequestBody.ActionEnum.fromValue("count"));
request.withBody(body);
try {
    ListTagResourceInstancesResponse response = client.listTagResourceInstances(request);
    System.out.println(response.toString());
} catch (ConnectionException e) {
    e.printStackTrace();
} catch (RequestTimeoutException e) {
    e.printStackTrace();
} catch (ServiceResponseException e) {
    e.printStackTrace();
    System.out.println(e.getHttpStatusCode());
    System.out.println(e.getRequestId());
    System.out.println(e.getErrorCode());
    System.out.println(e.getErrorMsg());
}
}
```

Python

- action为filter时，根据标签查询资源。

```
# coding: utf-8
```

```
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdc.v3.region.dc_region import DcRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdc.v3 import *
```

```
if __name__ == "__main__":
```

```
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
```


security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.

In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment

```
ak = os.getenv("CLOUD_SDK_AK")
sk = os.getenv("CLOUD_SDK_SK")

credentials = BasicCredentials(ak, sk) \

client = DcClient.new_builder() \
    .with_credentials(credentials) \
    .with_region(DcRegion.value_of("<YOUR REGION>")) \
    .build()

try:
    request = ListTagResourceInstancesRequest()
    listValuesNotTagsAny = [
        "value1",
        "value2"
    ]
    listNotTagsAnybody = [
        Tags(
            key="key1",
            values=listValuesNotTagsAny
        )
    ]
    listValuesTagsAny = [
        "value1",
        "value2"
    ]
    listTagsAnybody = [
        Tags(
            key="key1",
            values=listValuesTagsAny
        )
    ]
    listValuesTags = [
        "value1",
        "value2"
    ]
    listTagsbody = [
        Tags(
            key="key1",
            values=listValuesTags
        )
    ]
    listValuesNotTags = [
        "value1",
        "value2"
    ]
    listNotTagsbody = [
        Tags(
            key="key1",
            values=listValuesNotTags
        )
    ]
    listMatchesbody = [
        Match(
            key="resource_name",
            value="resource1"
        )
    ]
    request.body = ListTagResourceInstancesRequestBody(
        not_tags_any=listNotTagsAnybody,
        tags_any=listTagsAnybody,
        tags=listTagsbody,
        not_tags=listNotTagsbody,
        matches=listMatchesbody,
```

```
        action="filter",
        limit="10",
        offset="10"
    )
    response = client.list_tag_resource_instances(request)
    print(response)
except exceptions.ClientRequestException as e:
    print(e.status_code)
    print(e.request_id)
    print(e.error_code)
    print(e.error_msg)
```

- action为count时，查询资源总数。

```
# coding: utf-8
```

```
from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdc.v3.region.dc_region import DcRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdc.v3 import *

if __name__ == "__main__":
    # The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
    # security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
    # environment variables and decrypted during use to ensure security.
    # In this example, AK and SK are stored in environment variables for authentication. Before
    # running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local
    # environment
    ak = os.getenv("CLOUD_SDK_AK")
    sk = os.getenv("CLOUD_SDK_SK")

    credentials = BasicCredentials(ak, sk) \

    client = DcClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(DcRegion.value_of("<YOUR REGION>")) \
        .build()

    try:
        request = ListTagResourceInstancesRequest()
        listValuesSysTags = [
            "5aa119a8-d25b-45a7-8d1b-88e127885635"
        ]
        listSysTagsbody = [
            Tags(
                key="_sys_enterprise_project_id",
                values=listValuesSysTags
            )
        ]
        listValuesNotTagsAny = [
            "value1",
            "value2"
        ]
        listNotTagsAnybody = [
            Tags(
                key="key1",
                values=listValuesNotTagsAny
            )
        ]
        listValuesTagsAny = [
            "value1",
            "value2"
        ]
        listTagsAnybody = [
            Tags(
                key="key1",
                values=listValuesTagsAny
            )
        ]
        listValuesTags = [
            "value1",
```

```
    "value2"
  ]
  listValuesTags1 = [
    "value1",
    "value2"
  ]
  listTagsbody = [
    Tags(
      key="key1",
      values=listValuesTags1
    ),
    Tags(
      key="key2",
      values=listValuesTags
    )
  ]
  listValuesNotTags = [
    "value1",
    "*"value2"
  ]
  listNotTagsbody = [
    Tags(
      key="key1",
      values=listValuesNotTags
    )
  ]
  listMatchesbody = [
    Match(
      key="resource_name",
      value="resource1"
    )
  ]
  request.body = ListTagResourceInstancesRequestBody(
    sys_tags=listSysTagsbody,
    not_tags_any=listNotTagsAnybody,
    tags_any=listTagsAnybody,
    tags=listTagsbody,
    not_tags=listNotTagsbody,
    matches=listMatchesbody,
    action="count"
  )
  response = client.list_tag_resource_instances(request)
  print(response)
except exceptions.ClientRequestException as e:
  print(e.status_code)
  print(e.request_id)
  print(e.error_code)
  print(e.error_msg)
```

Go

- action为filter时，根据标签查询资源。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dc "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
    // security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
    // environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before
    // running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local
    // environment
```

```
ak := os.Getenv("CLOUD_SDK_AK")
sk := os.Getenv("CLOUD_SDK_SK")

auth := basic.NewCredentialsBuilder().
    WithAk(ak).
    WithSk(sk).
    Build()

client := dc.NewDcClient(
    dc.DcClientBuilder().
        WithRegion(region.ValueOf("<YOUR REGION>")).
        WithCredential(auth).
        Build())

request := &model.ListTagResourceInstancesRequest{}
var listValuesNotTagsAny = []string{
    "value1",
    "value2",
}
var listNotTagsAnybody = []model.Tags{
    {
        Key: "key1",
        Values: listValuesNotTagsAny,
    },
}
var listValuesTagsAny = []string{
    "value1",
    "value2",
}
var listTagsAnybody = []model.Tags{
    {
        Key: "key1",
        Values: listValuesTagsAny,
    },
}
var listValuesTags = []string{
    "*value1",
    "value2",
}
var listTagsbody = []model.Tags{
    {
        Key: "key1",
        Values: listValuesTags,
    },
}
var listValuesNotTags = []string{
    "*value1",
    "value2",
}
var listNotTagsbody = []model.Tags{
    {
        Key: "key1",
        Values: listValuesNotTags,
    },
}
var listMatchesbody = []model.Match{
    {
        Key: "resource_name",
        Value: "resource1",
    },
}
limitListTagResourceInstancesRequestBody:= "10"
offsetListTagResourceInstancesRequestBody:= "10"
request.Body = &model.ListTagResourceInstancesRequestBody{
    NotTagsAny: &listNotTagsAnybody,
    TagsAny: &listTagsAnybody,
    Tags: &listTagsbody,
    NotTags: &listNotTagsbody,
    Matches: &listMatchesbody,
```

```
Action: model.GetListTagResourceInstancesRequestBodyActionEnum().FILTER,
Limit: &limitListTagResourceInstancesRequestBody,
Offset: &offsetListTagResourceInstancesRequestBody,
}
response, err := client.ListTagResourceInstances(request)
if err == nil {
    fmt.Printf("%+v\n", response)
} else {
    fmt.Println(err)
}
}
```

- action为count时，查询资源总数。

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dc "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
    // security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
    // environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before
    // running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local
    // environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := dc.NewDcClient(
        dc.DcClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ListTagResourceInstancesRequest{}
    var listValuesSysTags = []string{
        "5aa119a8-d25b-45a7-8d1b-88e127885635",
    }
    var listSysTagsbody = []model.Tags{
        {
            Key: "_sys_enterprise_project_id",
            Values: listValuesSysTags,
        },
    }
    var listValuesNotTagsAny = []string{
        "value1",
        "value2",
    }
    var listNotTagsAnybody = []model.Tags{
        {
            Key: "key1",
            Values: listValuesNotTagsAny,
        },
    }
    var listValuesTagsAny = []string{
        "value1",
        "value2",
    }
    var listTagsAnybody = []model.Tags{
        {
```

```
        Key: "key1",
        Values: listValuesTagsAny,
    },
}
var listValuesTags = []string{
    "value1",
    "value2",
}
var listValuesTags1 = []string{
    "value1",
    "value2",
}
var listTagsbody = []model.Tags{
    {
        Key: "key1",
        Values: listValuesTags1,
    },
    {
        Key: "key2",
        Values: listValuesTags,
    },
}
var listValuesNotTags = []string{
    "value1",
    "*value2",
}
var listNotTagsbody = []model.Tags{
    {
        Key: "key1",
        Values: listValuesNotTags,
    },
}
var listMatchesbody = []model.Match{
    {
        Key: "resource_name",
        Value: "resource1",
    },
}
request.Body = &model.ListTagResourceInstancesRequestBody{
    SysTags: &listSysTagsbody,
    NotTagsAny: &listNotTagsAnybody,
    TagsAny: &listTagsAnybody,
    Tags: &listTagsbody,
    NotTags: &listNotTagsbody,
    Matches: &listMatchesbody,
    Action: model.GetListTagResourceInstancesRequestBodyActionEnum().COUNT,
}
response, err := client.ListTagResourceInstances(request)
if err == nil {
    fmt.Printf("%+v\n", response)
} else {
    fmt.Println(err)
}
```

更多

更多编程语言的SDK代码示例，请参见[API Explorer](#)的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
200	OK

错误码

请参见[错误码](#)。

5.5 配额管理

5.5.1 查询配额

功能介绍

查询租户各类资源的使用情况，如Directconnect的使用量，虚拟接口的使用量等。

调用方法

请参见[如何调用API](#)。

URI

GET /v3/{project_id}/dcaas/quotas

表 5-165 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	租户项目ID

表 5-166 Query 参数

参数	是否必选	参数类型	描述
type	否	Array	支持过滤的配额类型： <ul style="list-style-type: none">physicalConnect: 物理连接 direct_connect实例的配额和使用量virtualInterface: 虚拟接口 virtual-interface的配额和使用量globalDcGateway 专线全球接入网关的配额和使用量peerLinkPerGdgw: 接入网关的关联连接的配额和使用量

请求参数

无

响应参数

状态码： 201

表 5-167 响应 Body 参数

参数	参数类型	描述
quotas	quotas object	配额使用详情

表 5-168 quotas

参数	参数类型	描述
resources	Array of Info objects	不同资源配额使用详情列表

表 5-169 Info

参数	参数类型	描述
type	String	配额类型
quota	Long	可用的配额数，-1 代表不受限制
used	Long	已使用的配额数量
unit	String	用量单位

请求示例

查询配额信息

```
GET https://{dc_endpoint}/v3/08d5a9564a704afda6039ae2babbef3c/dcaas/quotas
```

响应示例

状态码： 201

查询配额成功

```
{
  "quotas": {
    "resources": [ {
      "type": "direct_connect",
      "quota": 1,
      "used": 0,
      "unit": "count"
    } ]
  }
}
```


SDK 代码示例

SDK代码示例如下。

Java

```
package com.huaweicloud.sdk.test;

import com.huaweicloud.sdk.core.auth.ICredential;
import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.exception.ConnectionException;
import com.huaweicloud.sdk.core.exception.RequestTimeoutException;
import com.huaweicloud.sdk.core.exception.ServiceResponseException;
import com.huaweicloud.sdk.dc.v3.region.DcRegion;
import com.huaweicloud.sdk.dc.v3.*;
import com.huaweicloud.sdk.dc.v3.model.*;

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

public class ShowQuotasSolution {

    public static void main(String[] args) {
        // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great
        // security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or
        // environment variables and decrypted during use to ensure security.
        // In this example, AK and SK are stored in environment variables for authentication. Before running
        // this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
        String ak = System.getenv("CLOUD_SDK_AK");
        String sk = System.getenv("CLOUD_SDK_SK");

        ICredential auth = new BasicCredentials()
            .withAk(ak)
            .withSk(sk);

        DcClient client = DcClient.newBuilder()
            .withCredential(auth)
            .withRegion(DcRegion.valueOf("<YOUR REGION>"))
            .build();
        ShowQuotasRequest request = new ShowQuotasRequest();
        request.withType();
        try {
            ShowQuotasResponse response = client.showQuotas(request);
            System.out.println(response.toString());
        } catch (ConnectionException e) {
            e.printStackTrace();
        } catch (RequestTimeoutException e) {
            e.printStackTrace();
        } catch (ServiceResponseException e) {
            e.printStackTrace();
            System.out.println(e.getHttpStatusCode());
            System.out.println(e.getRequestId());
            System.out.println(e.getErrorCode());
            System.out.println(e.getErrorMsg());
        }
    }
}
```

Python

```
# coding: utf-8

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkdc.v3.region.dc_region import DcRegion
from huaweicloudsdkcore.exceptions import exceptions
from huaweicloudsdkdc.v3 import *

if __name__ == "__main__":
```

```
# The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
# In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
ak = os.getenv("CLOUD_SDK_AK")
sk = os.getenv("CLOUD_SDK_SK")

credentials = BasicCredentials(ak, sk) \

client = DcClient.new_builder() \
    .with_credentials(credentials) \
    .with_region(DcRegion.value_of("<YOUR REGION>")) \
    .build()

try:
    request = ShowQuotasRequest()
    request.type =
    response = client.show_quotas(request)
    print(response)
except exceptions.ClientRequestException as e:
    print(e.status_code)
    print(e.request_id)
    print(e.error_code)
    print(e.error_msg)
```

Go

```
package main

import (
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    dc "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/model"
    region "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/dc/v3/region"
)

func main() {
    // The AK and SK used for authentication are hard-coded or stored in plaintext, which has great security risks. It is recommended that the AK and SK be stored in ciphertext in configuration files or environment variables and decrypted during use to ensure security.
    // In this example, AK and SK are stored in environment variables for authentication. Before running this example, set environment variables CLOUD_SDK_AK and CLOUD_SDK_SK in the local environment
    ak := os.Getenv("CLOUD_SDK_AK")
    sk := os.Getenv("CLOUD_SDK_SK")

    auth := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        Build()

    client := dc.NewDcClient(
        dc.DcClientBuilder().
            WithRegion(region.ValueOf("<YOUR REGION>")).
            WithCredential(auth).
            Build())

    request := &model.ShowQuotasRequest{}
    response, err := client.ShowQuotas(request)
    if err == nil {
        fmt.Printf("%+v\n", response)
    } else {
        fmt.Println(err)
    }
}
```

更多

更多编程语言的SDK代码示例，请参见[API Explorer](#)的代码示例页签，可生成自动对应的SDK代码示例。

状态码

状态码	描述
201	查询配额成功

错误码

请参见[错误码](#)。

6 公共参数

6.1 通用请求返回值

正常返回码	类型	说明
200	OK	GET和PUT操作正常返回。
201	Created	POST操作正常返回。
204	No Content	DELETE操作正常返回。

返回值	说明
400 Bad Request	服务器未能处理请求。
401 Unauthorized	被请求的页面需要用户名和密码。
403 Forbidden	对被请求页面的访问被禁止。
404 Not Found	服务器无法找到被请求的页面。
405 Method Not Allowed	请求中指定的方法不被允许。
406 Not Acceptable	服务器生成的响应无法被客户端所接受。
407 Proxy Authentication Required	用户必须首先使用代理服务器进行验证，这样请求才会被处理。
408 Request Timeout	请求超出了服务器的等待时间。
409 Conflict	由于冲突，请求无法被完成。
500 Internal Server Error	请求未完成。服务异常。
501 Not Implemented	请求未完成。服务器不支持所请求的功能。

返回值	说明
502 Bad Gateway	请求未完成。服务器从上游服务器收到一个无效的响应。
503 Service Unavailable	请求未完成。系统暂时异常。
504 Gateway Timeout	网关超时。

6.2 错误码

状态码	错误码	说明	处理措施
400	DC.0000	请求体异常。	输入正确参数重新下发请求。
400	DC.0001	请求参数异常。	输入正确参数重新下发请求。
400	DC.0002	操作或查找资源不存在。	输入正确参数重新下发请求。
400	DC.0003	服务系统内部错误。	输入正确参数重新下发请求。
400	DC.0004	ip地址不合法。	输入正确参数重新下发请求。
400	DC.0005	VPC接口调用异常。	输入正确参数重新下发请求。
400	DC.0006	MO接口调用异常。	输入正确参数重新下发请求。
400	DC.0007	VPC不存在。	输入正确参数重新下发请求。
400	DC.0008	bgp asn超出范围。	输入正确参数重新下发请求。
400	DC.0009	无效的Policy权限访问。	请联系技术支持处理。
400	DC.0010	VPC_VNI不存在。	请联系技术支持处理。
400	DC.0011	AGENT异常。	输入正确参数重新下发请求。
400	DC.0012	调用CBC接口异常。	请联系技术支持处理。
400	DC.0013	无权操作字段。	请联系技术支持处理。
400	DC.1000	物理连接带宽不足	请联系技术支持处理。
400	DC.1001	创建非托管物理连接指定VLAN	请联系技术支持处理。

状态码	错误码	说明	处理措施
400	DC.1002	创建自动匹配托管专线失败	请联系技术支持处理。
400	DC.1003	创建托管物理连接未指定VLAN	请联系技术支持处理。
400	DC.1004	创建托管物理连接指定非运营物理连接	请联系技术支持处理。
400	DC.1005	物理连接状态异常	请联系技术支持处理。
400	DC.1006	使用中物理连接无法修改设备及类型	请联系技术支持处理。
400	DC.1007	物理连接使用中	请联系技术支持处理。
400	DC.1008	托管专线VLAN冲突	请联系技术支持处理。
400	DC.1010	专线带宽减容	请联系技术支持处理。
400	DC.1011	专线带宽不可修改	请联系技术支持处理。
400	DC.1012	物理连接不存在	请联系技术支持处理。
400	DC.1014	非标准专线设置冗余专线	请联系技术支持处理。
400	DC.1015	专线或者LAG不存在	请联系技术支持处理。
400	DC.1017	托管专线vlan为0	请联系技术支持处理。
400	DC.1019	物理连接配额已满	请联系技术支持处理。
400	DC.1100	虚拟网关本端endpoint group id非法	请联系技术支持处理。
400	DC.1101	本段子网数量超限	请联系技术支持处理。
400	DC.1102	未指定主设备时指定备设备	请联系技术支持处理。
400	DC.1103	主备设备相同	请联系技术支持处理。
400	DC.1104	已有device虚拟网关不可更新device信息	请联系技术支持处理。
400	DC.1105	本端CIDR与远端CIDR存在交集	请联系技术支持处理。
400	DC.1106	虚拟网关使用中	请联系技术支持处理。
400	DC.1107	虚拟网关VLAN耗尽	请联系技术支持处理。
400	DC.1108	虚拟网关VNI耗尽	请联系技术支持处理。
400	DC.1109	虚拟网关VRF耗尽	请联系技术支持处理。
400	DC.1110	VPC下已有虚拟网关	请联系技术支持处理。
400	DC.1111	虚拟网关不存在	请联系技术支持处理。

状态码	错误码	说明	处理措施
400	DC.1112	虚拟网关主备设备不构成主备关系	请联系技术支持处理。
400	DC.1113	虚拟网关处于割接状态不允许更新	请联系技术支持处理。
400	DC.1114	vlan或者单vtep设备的虚拟网关不支持更新traffic_mode	请联系技术支持处理。
400	DC.1115	VGW的vni不存在	请联系技术支持处理。
400	DC.1116	更新route gateway失败	请联系技术支持处理。
400	DC.1117	虚拟网关已关联了两条物理专线，无法关联更多	请联系技术支持处理。
400	DC.1118	VGW正在被操作（锁定）	请联系技术支持处理。
400	DC.1119	route gateway device group 创建失败	请联系技术支持处理。
400	DC.1120	main_az_list配置项有误	请联系技术支持处理。
400	DC.1200	虚拟接口未指定LAG或物理连接	请联系技术支持处理。
400	DC.1201	物理连接及虚拟网关类型不匹配	请联系技术支持处理。
400	DC.1202	double ipsec类型虚拟接口不支持创建虚拟接口	请联系技术支持处理。
400	DC.1203	BGP模式虚拟接口未指定AS号	请联系技术支持处理。
400	DC.1204	运营专线不支持创建虚拟接口	请联系技术支持处理。
400	DC.1205	创建虚拟接口关联资源状态异常	请联系技术支持处理。
400	DC.1206	托管专线已存在关联虚拟接口	请联系技术支持处理。
400	DC.1207	虚拟接口VLAN与托管专线不一致	请联系技术支持处理。
400	DC.1208	托管专线虚拟接口不可修改带宽	请联系技术支持处理。
400	DC.1209	虚拟接口VLAN冲突	请联系技术支持处理。
400	DC.1210	不可修改虚拟接口状态	请联系技术支持处理。
400	DC.1211	虚拟接口不存在	请联系技术支持处理。
400	DC.1212	虚拟网关与物理连接group不同	请联系技术支持处理。
400	DC.1213	虚拟接口和租户不匹配	请联系技术支持处理。

状态码	错误码	说明	处理措施
400	DC.1214	虚拟接口已经绑定	请联系技术支持处理。
400	DC.1215	虚拟接口类型不匹配	请联系技术支持处理。
400	DC.1216	待关联或待解除关联的本地网关无效	请联系技术支持处理。
400	DC.1217	物理连接ies id与lgw ies id不匹配	请联系技术支持处理。
400	DC.1218	lgw不匹配	请联系技术支持处理。
400	DC.1219	已有虚拟接口创建VLAN 0虚拟接口	请联系技术支持处理。
400	DC.1220	公网vif email字段为空	请联系技术支持处理。
400	DC.1221	所选虚拟网关设备不支持接入点接入	请联系技术支持处理。
400	DC.1222	POP vni不足	请联系技术支持处理。
400	DC.1223	本端BGP ASN与对端相同	请联系技术支持处理。
400	DC.1224	接入点接入专线虚拟接口不支持VLAN 0	请联系技术支持处理。
400	DC.1225	使能流统的专线虚拟接口超过阈值	请联系技术支持处理。
400	DC.1226	远端子接口（二层）的无子接口的接口不支持使能流统	请联系技术支持处理。
400	DC.1227	vif状态异常不允许更新流统	请联系技术支持处理。
400	DC.1400	端口组数量超限	请联系技术支持处理。
400	DC.1401	重复CIDR	请联系技术支持处理。
400	DC.1402	端口组使用中	请联系技术支持处理。
400	DC.1403	端口组不存在	请联系技术支持处理。

A 附录

A.1 获取项目 ID

操作场景

在调用接口的时候，部分URL中需要填入项目ID，所以需要获取到项目ID。有如下两种获取方式：

- [调用API获取项目ID](#)
- [从控制台获取项目ID](#)

调用 API 获取项目 ID

项目ID可以通过调用[查询指定条件下的项目列表](#)API获取。

获取项目ID的接口为“GET https://{Endpoint}/v3/projects”，其中{Endpoint}为IAM的终端节点，可以从[地区和终端节点](#)获取。接口的认证鉴权请参见[认证鉴权](#)。

响应示例如下，其中projects下的“id”即为项目ID。

```
{
  "projects": [
    {
      "domain_id": "65ewtrgaggshhk1223245sghjlse684b",
      "is_domain": false,
      "parent_id": "65ewtrgaggshhk1223245sghjlse684b",
      "name": "project_name",
      "description": "",
      "links": {
        "next": null,
        "previous": null,
        "self": "https://www.example.com/v3/projects/a4adasfjljaaaakla12334jklga9sasfg"
      },
      "id": "a4adasfjljaaaakla12334jklga9sasfg",
      "enabled": true
    }
  ],
  "links": {
    "next": null,
    "previous": null,
    "self": "https://www.example.com/v3/projects"
  }
}
```

从控制台获取项目 ID

从控制台获取项目ID的步骤如下：

1. 登录管理控制台。
2. 鼠标悬停在右上角的用户名，选择下拉列表中的“我的凭证”。
在“API凭证”页面的项目列表中查看项目ID。

图 A-1 查看项目 ID



B 修订记录

发布日期	修改记录
2022-11-21	第一次正式发布。