

视频点播

API 参考

文档版本 01
发布日期 2025-02-11



版权所有 © 华为云计算技术有限公司 2025。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云计算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，华为云计算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。

华为云计算技术有限公司

地址：贵州省贵安新区黔中大道交兴功路华为云数据中心 邮编：550029

网址：<https://www.huaweicloud.com/>

目录

1 使用前必读	1
2 API 概览	5
3 如何调用 API	9
3.1 构造请求	9
3.2 认证鉴权	12
3.3 返回结果	14
4 应用示例	16
4.1 示例 1: 媒资上传 (20M 以下)	16
4.2 示例 2: 媒资分段上传 (20M 以上)	21
4.3 示例 3: 获取媒资详情	31
4.4 示例 4: 视频处理	35
5 媒资上传	38
5.1 创建媒资: 上传方式	38
5.2 获取分段上传授权	46
5.3 确认媒资上传	50
5.4 桶授权	52
5.5 创建媒资: OBS 转存方式	54
5.6 创建媒资: URL 拉取注入 (公测中)	62
5.7 上传检验	69
6 媒资处理	72
6.1 视频更新	72
6.2 媒资处理	76
6.3 取消媒资转码任务	80
6.4 音频提取	82
6.5 取消提取音频任务	85
6.6 创建审核媒资任务	86
6.7 设置封面	91
7 媒资管理	93
7.1 删除媒资	93
7.2 媒资发布	95
7.3 媒资发布取消	105

7.4 查询媒资信息.....	115
7.5 修改媒资属性.....	128
7.6 查询指定媒资的详细信息.....	130
7.7 查询媒资列表.....	144
8 媒资预热.....	151
8.1 CDN 预热.....	151
8.2 查询 CDN 预热.....	153
9 媒资分类.....	157
9.1 创建媒资分类.....	157
9.2 修改媒资分类.....	160
9.3 删除媒资分类.....	162
9.4 查询指定分类信息.....	164
10 密钥查询.....	167
10.1 查询密钥.....	167
11 统计分析.....	170
11.1 查询 CDN 统计信息.....	170
11.2 查询源站统计信息.....	174
11.3 查询 TopN 媒资信息.....	178
11.4 查询域名播放日志.....	181
11.5 查询媒资日播放统计数据.....	183
12 水印模板管理.....	187
12.1 创建水印模板.....	187
12.2 修改水印模板.....	191
12.3 查询水印列表.....	195
12.4 删除水印模板.....	199
12.5 确认水印图片上传.....	200
13 字幕管理.....	203
13.1 多字幕封装.....	203
14 转码模板管理.....	207
14.1 创建自定义转码模板.....	207
14.2 查询转码模板列表.....	213
14.3 修改转码模板.....	220
14.4 删除自定义模板.....	226
15 转码模板集合管理.....	229
15.1 创建转码模板组集合.....	229
15.2 修改转码模板组集合.....	231
15.3 查询自定义模板组集合.....	233
15.4 删除转码模板组集合.....	239
16 转码产物管理.....	241

16.1 删除转码产物.....	241
17 媒资存储模式管理.....	246
17.1 修改媒资降冷粒度.....	246
17.2 修改媒资文件在 OBS 的存储模式.....	248
17.3 查询媒资降冷配置.....	251
17.4 查询取回数据信息.....	253
18 历史 API.....	257
18.1 媒资上传.....	257
18.1.1 获取分段上传授权.....	257
19 附录.....	261
19.1 状态码.....	261
19.2 错误码.....	262
19.3 获取项目 ID.....	272
19.4 获取账号 ID.....	274
19.5 生成 MD5 值.....	275
19.6 日志管理.....	276
19.7 分段上传代码示例.....	277
19.7.1 Java.....	277
19.7.2 POM 文件.....	284
19.7.3 JavaScript.....	284
19.7.4 Python.....	294
19.7.5 Go.....	300
19.7.6 PHP.....	308
20 修订记录.....	315

1 使用前必读

视频点播服务（Video on Demand）是集视频上传、自动化转码处理、媒体资源管理、分发加速、视频播放于一体的一站式解决方案。借助华为云提供灵活弹性解决方案，您无需关注服务依赖的底层基础设施，只需要依托高质量的媒体处理服务来快速搭建安全、弹性的点播平台。

您可以使用本文档提供API对视频点播进行相关操作，如音视频上传、转码、分类、管理等。支持的全部操作请参见[API概览](#)。

在调用视频点播API之前，请确保已经充分了解视频点播相关概念，详细信息请参见[产品介绍](#)。

终端节点

视频点播提供了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](#)。

终端节点即调用API的[请求地址](#)，不同区域的终端节点不同，您可以从[地区和终端节点](#)中查询服务的终端节点。

同时视频点播还提供多种编程语言的SDK供您使用，SDK的使用方法请参见[VOD SDK](#)。

支持跨域请求的接口说明

目前视频点播服务仅以下接口支持浏览器跨域请求，如[表1-1](#)所示。

表 1-1 支持跨域请求接口列表

接口名称	URI
创建媒资：上传方式	/v1.0/{project_id}/asset
确认媒资上传	/v1.0/{project_id}/asset/status/uploaded
上传检验	/v1.0/{project_id}/asset/duplication
查询媒资信息	/v1.0/{project_id}/asset/info
获取分段上传授权	/v1.0/{project_id}/asset/authority

API 流控

视频点播服务设置了API流控限制，避免出现短时间内重复调用API，服务中断的情况。API流控限制说明如表1-2所示。

表 1-2 API 流控限制

接口分类	接口名称	单租户接口流控 (单位: 次数/ 分钟)	接口总体流控 (单位: 次数/ 分钟)
媒资上传	<ul style="list-style-type: none"> 创建媒资: 上传方式 获取分段上传授权 确认媒资上传 桶授权 创建媒资: OBS转存方式 	1500	12000
	<ul style="list-style-type: none"> 创建媒资: URL拉取注入 上传检验 	100	1000
媒资处理	<ul style="list-style-type: none"> 视频更新 媒资处理 取消媒资转码任务 音频提取 取消提取音频任务 创建审核媒资任务 设置封面 	100	1000
媒资管理	<ul style="list-style-type: none"> 删除媒资 媒资发布 媒资发布取消 修改媒资属性 	100	1000
	查询媒资信息	1500	24000
	<ul style="list-style-type: none"> 查询指定媒资的详细信息 查询媒资列表 	1500	12000
媒资预热	<ul style="list-style-type: none"> CDN预热 查询CDN预热 	100	1000

接口分类	接口名称	单租户接口流控 (单位: 次数/ 分钟)	接口总体流控 (单位: 次数/ 分钟)
媒资分类	<ul style="list-style-type: none"> 创建媒资分类 修改媒资分类 删除媒资分类 查询指定分类信息 	100	1000
密钥查询	密钥查询	1500	12000
统计分析	<ul style="list-style-type: none"> 查询CDN统计信息 查询源站统计信息 查询TopN媒资信息 查询域名播放日志 查询媒资日播放统计数据 	100	1000
水印模板管理	<ul style="list-style-type: none"> 创建水印模板 修改水印模板 查询水印列表 删除水印模板 确认水印图片上传 	100	1000
多字幕封装	多字幕封装	100	1000
转码模板管理	<ul style="list-style-type: none"> 创建自定义转码模板 查询转码模板列表 修改转码模板 删除自定义模板 	100	1000
转码模板集合管理	<ul style="list-style-type: none"> 创建转码模板组集合 修改转码模板组集合 查询自定义模板组集合 删除转码模板组集合 	100	1000
转码产物管理	删除转码产物	100	1000
媒资存储模式管理	<ul style="list-style-type: none"> 修改媒资降冷粒度 修改媒资文件在OBS的存储模式 查询媒资降冷配置 	100	1000

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创建的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）。

在“我的凭证”下，您可以查看账号ID和用户ID。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

指云资源所在的物理位置，同一区域内可用区间内网互通，不同区域间内网不互通。通过在不同地区创建云资源，可以将应用程序设计的更接近特定客户的要求，或满足不同地区的法律或其他要求。

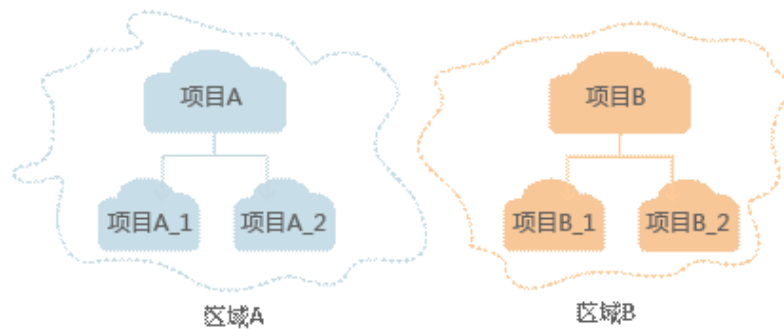
- 可用区

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，并在子项目中购买资源，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资源，使得资源的权限控制更加精确。

图 1-1 项目隔离模型



2 API 概览

视频点播服务对应的接口列表如下所示，在调用点播API前，您需要先获取用户Token，Token可以用于调用其他API时鉴权，具体如何调用点播API请参考[应用示例](#)。

媒资上传

接口	说明
POST /v1.0/{project_id}/asset	创建媒资：上传方式
GET /v1.1/{project_id}/asset/authority	获取分段上传授权
POST /v1.0/{project_id}/asset/status/uploaded	确认媒资上传
PUT /v1.0/{project_id}/asset/authority	桶授权
POST /v1.0/{project_id}/asset/reproduction	创建媒资：OBS转存方式
POST /v1.0/{project_id}/asset/upload_by_url	创建媒资：URL拉取注入
GET /v1.0/{project_id}/asset/duplication	上传检验

媒资处理

接口	说明
PUT /v1.0/{project_id}/asset	视频更新
POST /v1.0/{project_id}/asset/process	媒资处理
DELETE /v1.0/{project_id}/asset/process	取消媒资转码任务
POST /v1.0/{project_id}/asset/extract_audio	音频提取
DELETE /v1.0/{project_id}/asset/extract_audio	取消提取音频任务
PUT /v1.0/{project_id}/asset/cover	设置封面

媒资管理

接口	说明
DELETE /v1.0/{project_id}/asset	删除媒资
POST /v1.0/{project_id}/asset/status/publish	媒资发布
POST /v1.0/{project_id}/asset/status/unpublish	媒资发布取消
GET /v1.0/{project_id}/asset/info	查询媒资信息
PUT /v1.0/{project_id}/asset/info	修改媒资属性
GET /v1.0/{project_id}/asset/details	查询指定媒资的详细信息
GET /v1.0/{project_id}/asset/list	查询媒资列表

媒资预热

接口	说明
POST /v1.0/{project_id}/asset/preheating	CDN预热
GET /v1.0/{project_id}/asset/preheating	查询CDN预热

媒资分类

接口	说明
POST /v1.0/{project_id}/asset/category	创建媒资分类
PUT /v1.0/{project_id}/asset/category	修改媒资分类
DELETE /v1.0/{project_id}/asset/category	删除媒资分类
GET /v1.0/{project_id}/asset/category	查询指定分类信息

密钥查询

接口	说明
GET /v1.0/{project_id}/asset/ciphers	查询密钥

统计分析

接口	说明
GET /v1.0/{project_id}/asset/cdn-statistics	查询CDN统计信息

接口	说明
GET /v1.0/{project_id}/asset/vod-statistics	查询源站统计信息
GET /v1.0/{project_id}/asset/top-statistics	查询TopN视频信息
GET /v1.0/{project_id}/vod/cdn/logs	查询域名播放日志
GET /v1/{project_id}/asset/daily-summary	查询媒资日播放统计数据

水印模板管理

接口	说明
POST /v1.0/{project_id}/template/watermark	创建水印模板
PUT /v1.0/{project_id}/template/watermark	修改水印模板
GET /v1.0/{project_id}/template/watermark	查询水印列表
DELETE /v1.0/{project_id}/template/watermark	删除水印模板
POST /v1.0/{project_id}/watermark/status/uploaded	确认水印图片上传

多字幕封装

接口	说明
PUT /v1/{project_id}/asset/subtitles	多字幕封装

转码模板管理

接口	说明
POST /v2/{project_id}/asset/template/transcodings	创建自定义转码模板
GET /v2/{project_id}/asset/template/transcodings	查询转码模板列表
PUT /v2/{project_id}/asset/template/transcodings	修改转码模板
DELETE /v2/{project_id}/asset/template/transcodings	删除自定义模板

转码模板集合管理

接口	说明
POST /v1.0/{project_id}/asset/template-collection/transcodings	创建转码模板组集合
PUT /v1.0/{project_id}/asset/template-collection/transcodings	修改转码模板组集合
GET /v1.0/{project_id}/asset/template-collection/transcodings	查询自定义模板组集合
DELETE /v1.0/{project_id}/asset/template-collection/transcodings	删除转码模板组集合

转码产物管理

接口	说明
DELETE /v1/{project_id}/asset/transcode-product	删除转码产物

媒资存储模式管理

接口	说明
PUT /v1/{project_id}/asset/storage-mode-type	修改媒资降冷粒度
PUT /v1/{project_id}/asset/storage-mode	修改媒资文件在OBS的存储模式
GET /v1/{project_id}/asset/storage-mode-type	查询媒资降冷配置

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。

例如您需要您创建IAM用户，由于IAM为全局服务，则使用任一区域的Endpoint，比如“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/v3.0/OS-USER/users
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[状态码](#)。

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 应用示例

4.1 示例 1：媒资上传（20M 以下）

场景描述

若您需要将存储在本地的音视频文件上传到点播服务，并在上传后，自动处理音视频，如发布、转码等，您可以调用VOD API进行媒资创建，并通过OBS API将音视频文件上传到VOD的存储桶中。

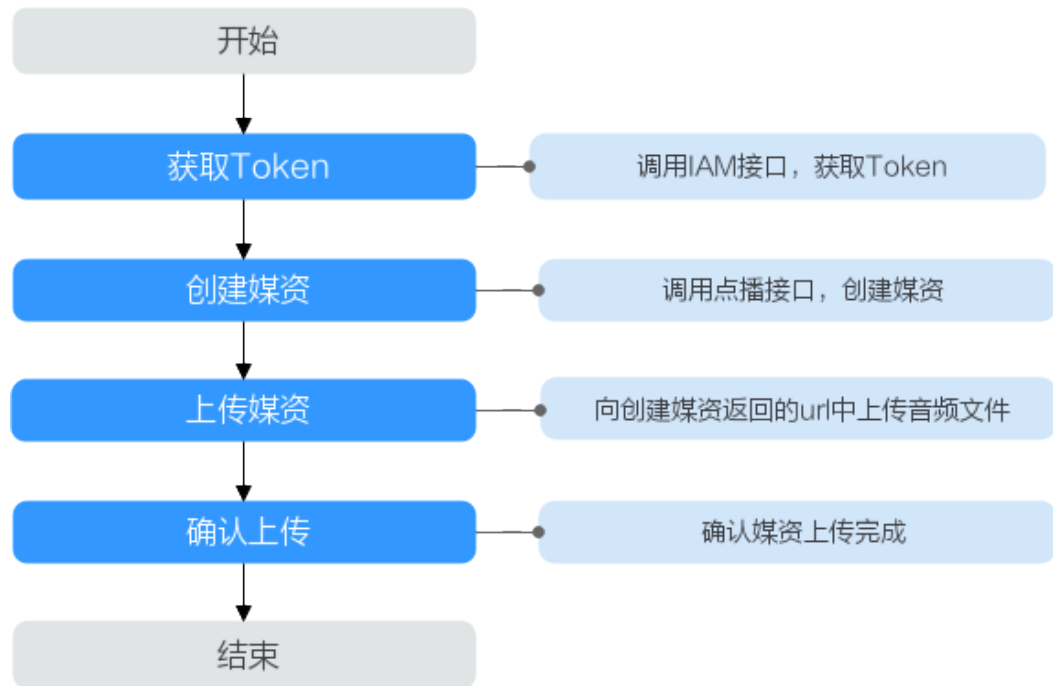
本章节指导用户如何使用API调用的方式上传一个大小20M以下的视频文件。

前提条件

- 已明确需要上传媒资的点播服务所在的区域信息，详见[使用前必读](#)。
- 已获取需要上传到对应点播服务所在区域的项目ID，具体获取方法请参见[获取项目ID](#)。

总体流程

图 4-1 媒资上传流程



1. 调用**创建媒资：上传方式**接口创建媒资。
2. 使用PUT方法将本地媒资文件上传。
3. 调用**确认媒资上传**接口完成媒资创建确认。

操作步骤

步骤1 获取用户Token，用于调用点播接口时进行认证鉴权。

详细信息请参见**构造请求**。以“华北-北京四”为示例，若您需要调用其它区域的点播接口，请替换成对应区域的**IAM终端节点**。

```
POST https://iam.cn-north-4.myhuaweicloud.com/v3/auth/tokens
Content-Type: application/json
```

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

```
}
}
}
}
```

如图4-2所示，红框中的信息为获取用户Token。

图 4-2 获取用户 Token

```
cache-control → no-cache, no-store, must-revalidate
connection → keep-alive
content-length → 15185
content-type → application/json; charset=UTF-8
date → Wed, 29 May 2019 08:02:38 GMT
expires → Thu, 01 Jan 1970 00:00:00 GMT
pragma → no-cache
server → api-gateway
strict-transport-security → max-age=31536000; includeSubdomains;
via → proxy A
x-content-type-options → nosniff
x-download-options → noopen
x-frame-options → SAMEORIGIN
x-iam-trace-id → token_global_null_2fed19b42067d0b94ae73d8e0b8b7841
x-request-id → fd8e4d01ae9ca9f58af20c8db2c88e2
x-subject-token → MIIYdWVJKoZlhcNAQcCgllYdCCGQCAQeXDTALBgIghg8ZQMfAgEwghbF8gkqhkG9w08BwGggha28llWnsidG9rZW4OnsiZxhwaKlc19hdCl6JwMTktMDUHMzBUMDg6MDI6MzguNTgxMDAwWlslm1ldGhvZHMlOlscGfzc3dvcnQXSwYzF0YWVwZy5lYm9weSwjRGlY3xupf0OauVL-E275QpmEOPG4qKpRfIeeZiH6Zcfp85A2L-RGdrdWLE95IeegpR0PGK9zpbuRwYAtIdQL3xGvVCLUm8XJADnwO2fWygbmzrPwFbhp3F20TK4v9XhwWHLxH2yfo+KScgsXHAz6VURCWMdV+UeE5wZo7vR22C77p289d88lyk3eQzQmTyrIaupZ6I4PX2J+HW44plwEszdvrRQLLYOzxinGG3AF
x-xss-protection → 1; mode=block
```

步骤2 调用**创建媒资：上传方式**接口，创建媒资，在请求头中增加“X-Auth-Token”字段，其中“Value”为**步骤1**获取的。

POST https://vod.cn-north-4.myhuaweicloud.com/v1.0/{project_id}/asset

```
{
  "title": "test",
  "description": "test",
  "category_id": 87748,
  "video_name": "test.mp4",
  "video_type": "MP4",
  "cover_type": "PNG",
  "tags": "mytags,test",
  "auto_publish": 0,
  "template_group_name": "test",
  "subtitles": [
    {
      "id": 1,
      "language": "CN",
      "type": "SRT",
      "md5": "SqcyFJJZoDZaP8oKIY6rgQ==",
      "description": "AAAAA",
      "name": "cc.srt"
    }
  ],
  "thumbnail": {
    "type": "time",
    "cover_position": 1
  },
  "review": {
    "interval": 10,
    "politics": -1,
    "terrorism": -1,
    "porn": -1
  }
}
```

- “vod.cn-north-4.myhuaweicloud.com”为点播服务的终端节点，您可以在**VOD 终端节点**中获取。

- “category_id” 为媒资分类ID。用户需要提前创建完成媒资分类，操作请参考[创建媒资分类](#)。
- “template_group_name” 为转码模板组名称。用户需要提前创建完成转码模板，操作请参考[创建自定义转码模板](#)。

说明

调用IAM接口的区域需与调用点播接口的区域一致，如调用“cn-north-4”区域的IAM接口获取用户Token，则该Token仅能用于调用“cn-north-4”区域点播接口的鉴权。

步骤3 在返回参数中获取 “video_upload_url”，“cover_upload_url” 和 “subtitle_upload_urls”。

```
{
  "asset_id": "3f40a2c7c60454f5f84381e0313ca230",
  "video_upload_url": "https://vod-bucket-81.obs.cn-north-4.myhuaweicloud.com:443/474bcff2992f4be5b883a2fb9cec7343/3f40a2c7c60454f5f84381e0313ca230/cdeda86cd1b7b3dd760a3ff28a5ee497.mp4?AWSAccessKeyId=BG923RWHL4HFXOGKCVL&Expires=1560420274&Signature=9f%2BZcdD6SwjlU5ARHYiP6YY1Lyw%3D",
  "cover_upload_url": "https://vod-bucket-81.obs.cn-north-4.myhuaweicloud.com:443/474bcff2992f4be5b883a2fb9cec7343/3f40a2c7c60454f5f84381e0313ca230/cover/Cover0.png?AWSAccessKeyId=BG923RWHL4HFXOGKCVL&Expires=1560420274&Signature=4Aa88NK%2By%2By1Xo0RvLpOvuaFCoE%3D",
  "subtitle_upload_urls": [
    "https://vod-bucket-81.obs.cn-north-4.myhuaweicloud.com:443/474bcff2992f4be5b883a2fb9cec7343/3f40a2c7c60454f5f84381e0313ca230/subtitle/1.srt?AWSAccessKeyId=BG923RWHL4HFXOGKCVL&Expires=1560420274&Signature=l0UclE9yfaVrxkl0kaNnr%2BemG98%3D"
  ],
  "target": {
    "bucket": "vod-bucket-81",
    "location": "cn-north-4",
    "object": "474bcff2992f4be5b883a2fb9cec7343/3f40a2c7c60454f5f84381e0313ca230/cdeda86cd1b7b3dd760a3ff28a5ee497.mp4"
  }
}
```

步骤4 使用PUT请求分别向 “video_upload_url”，“cover_upload_url” 和 “subtitle_upload_urls” 中上传音视频文件、封面图片和字幕文件。

其中，请求头 “Content-Type” 需要根据上传的文件类型分别设置，如下所示：

- 视频文件：video/*视频格式*，如：“video/mp4”。更多视频文件格式对应的请求头填写规则，请参见[表4-1](#)。
- 音频文件：audio/*音频格式*，如：“audio/mp3”。更多音频文件格式对应的请求头填写规则，请参见[表4-2](#)。
- 图片文件：image/*图片格式*，如：“image/png”。
- 字幕文件：application/octet-stream。

表 4-1 视频文件请求头填写说明

文件后缀	Content-Type
MP4	video/mp4
MOV	video/quicktime
MXF	application/mxf

文件后缀	Content-Type
TS	video/mp2t
MPG	video/mpeg
FLV	video/flv
WMV	video/x-ms-wmv
AVI	video/x-msvideo
M4V	video/m4v
F4V	application/f4v
MPEG	video/mpeg
M3U8	application/octet-stream
_3GP/3GP	video/3gpp
ASF	video/x-ms-asf
MKV	video/x-matroska
WEBM	video/webm
MPD	video/dash

表 4-2 音频文件请求头填写说明

文件后缀	Content-Type
MP3	audio/mp3
WMA	audio/wma
APE	audio/ape
FLAC	audio/flac
AAC	audio/aac
AC3	audio/ac3
MMF	audio/mmf
AMR	audio/amr
M4A	audio/m4a
M4R	audio/m4r
OGG	audio/ogg
WAV	audio/wav

文件后缀	Content-Type
WV	audio/wv
MP2	audio/mp2

步骤5 调用接口[确认媒资上传](#)接口，完成媒资上传。

```
POST https://vod.cn-north-4.myhuaweicloud.com/v1.0/{project_id}/asset/status/uploaded
{
  "asset_id": "0f4d3f1f32ec353d8866f2d84a036124",
  "status": "CREATED"
}
```

----**结束**

媒资上传成功后，您可以在视频点播控制台查看媒资相关详细信息。

4.2 示例 2：媒资分段上传（20M 以上）

场景描述

示例1：媒资上传（20M以下）已介绍了如何将小于20M大小的音视频文件上传到点播服务。若您需要上传的音视频文件的大小大于20M，您可以调用VOD API进行分段上传。

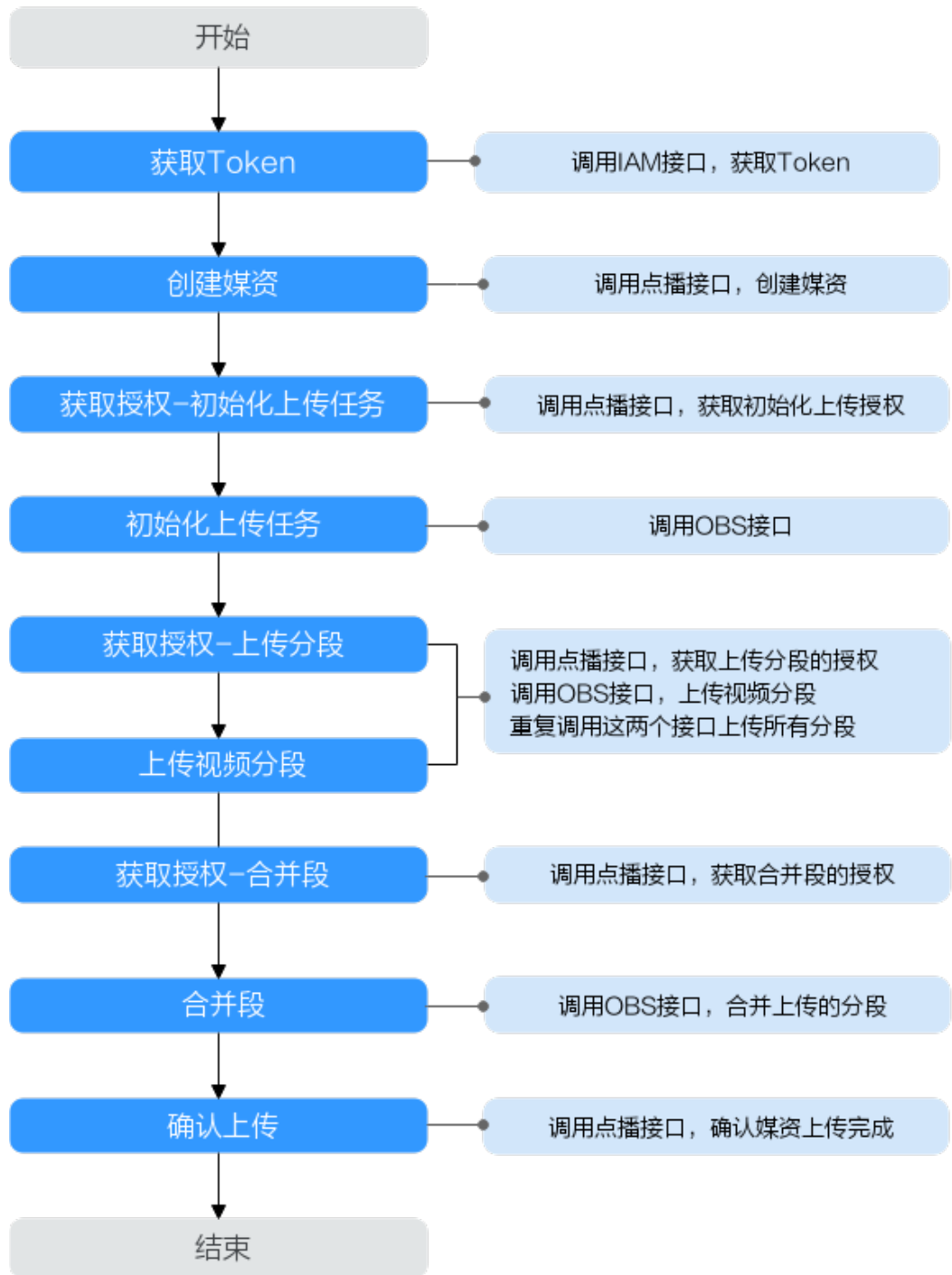
本章节指导用户如何使用API调用的方式上传一个文件大小大于20M的视频文件。

前提条件

- 已明确需要上传媒资的点播服务所在的区域信息，详见[使用前必读](#)。
- 已获取需要上传到对应点播服务所在区域的项目ID，具体获取方法请参见[获取项目ID](#)。
- 待上传的视频文件已完成二进制流分割，建议每个分段的大小不超过20M。
- OBS接口[列举已上传的段](#)支持响应段的取值范围[1,10000]，段大小[100KB,5GB]。默认最多响应1000段，如果用户分段数目超出1000，需参考OBS接口[列举已上传的段](#)进行处理。

总体流程

图 4-3 媒资上传流程



1. [获取用户Token](#)
2. [创建点播媒资](#)
3. [获取初始化上传任务的授权](#)
4. [初始化上传任务](#)

5. [获取上传分段的授权](#)
6. [上传分段](#)
7. 参考[5-6](#)，将所有视频分段上传
8. [获取列举已上传分段的授权](#)
9. [获取已上传的分段](#)
10. [获取合并段的授权](#)
11. [合并上传的分段](#)
12. [确认媒资上传](#)

操作步骤

步骤1 获取用户Token，用于调用点播接口时进行认证鉴权。

详细信息请参见[构造请求](#)。以“华北-北京四”为例，若您需要调用其它区域的点播接口，请替换成对应区域的[IAM终端节点](#)。

POST <https://iam.cn-north-4.myhuaweicloud.com/v3/auth/tokens>
Content-Type: application/json

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

如[图4-4](#)所示，红框中的信息为获取用户Token。

图 4-4 获取用户 Token

```

cache-control → no-cache, no-store, must-revalidate
connection → keep-alive
content-length → 15185
content-type → application/json; charset=UTF-8
date → Wed, 29 May 2019 08:02:38 GMT
expires → Thu, 01 Jan 1970 00:00:00 GMT
pragma → no-cache
server → api-gateway
strict-transport-security → max-age=31536000; includeSubdomains;
via → proxy A
x-content-type-options → nosniff
x-download-options → noopen
x-frame-options → SAMEORIGIN
x-iam-trace-id → token_global_null_2fed19b42067d0b94ae73dbe0b8b7841
x-request-id → f48e4d01ae9ca9f5af20cb2c88e2
x-subject-token
→ MllYdwYJkoZlthvcNAQc:ColYadCCGQCAQExDTALBglghkgBQMEAgEwghbFgkqhkiG9w0BBwGgggha2BllWnsrdG9:ZW4:OmsiZxhwaXk:19hdCl6JwMTktMDUHMzBUMDg6MDI6MzguNTgxDwWlslm1ldGhvZHMlOisicGfzc3dvcnQzXSwyZf0YWwZy:
wN-yNDwpsWwyRGTY3upfYOauVL-E275QpmEOPG4qKzPv4YeoZH8Zdfpt86AZt:
KGDroWLe5ShmgzR0fGKzgbuRWYA1qQL3xGvVcUUm8KlADmwOZTWgbrmzPwFbVhP3f20TK4v9KhvWHUjwH2yfo+KScgpXHAzveVURCIWMDV+UeESwZa7v82CZ7p289d88y3oQz2mTylaupZf6J4PX2I+rW4aphwEszdrRQLLYOzsrGG3Af
x-ssr-protection → 1; mode=block
    
```

步骤2 调用**创建媒资：上传方式**接口，创建媒资。

1. 选择**POST**请求方法并输入URI。
https://vod.cn-north-4.myhuaweicloud.com/v1.0/{project_id}/asset
2. 请求消息头中增加“X-Auth-Token”，值设置为**步骤1**中获取的用户Token。
3. Request Body中传入参数如下：

```

{
  "title": "testVideo",
  "description": "test",
  "category_id": 87748,
  "video_name": "test.mp4",
  "video_type": "MP4",
  "auto_publish": 0
}
    
```

说明

示例仅设置了部分媒资参数，您可以根据实际需求参考**创建媒资：上传方式**接口的请求参数设置。

4. 请求响应成功后，返回处理媒资ID、视频上传地址及存储视频文件的OBS桶信息。

```

{
  "asset_id": "fea62a2845e43c37b4cbfb017c0d0821",
  "video_upload_url": "https://vod-bucket-65-cn-north-4.obs.cn-north-4.myhuaweicloud.com:443/05041ffa4002****f6dc009cc6f8f33/fea62a2845e43c37b4cbfb017c0d0821/d1f221f09f6bfffefb882c8f9e167483a.mp4?AWSAccessKeyId=MZH0LUL329N1SSXNB3S4&Expires=1596700132&Signature=GIHTcO2avO9B84sH8A5wL2GieCl%3D",
  "cover_upload_url": null,
  "subtitle_upload_urls": [],
  "target": {
    "bucket": "vod-bucket-65-cn-north-4",
    "location": "cn-north-4",
    "object": "05041ffa4002****f6dc009cc6f8f33/fea62a2845e43c37b4cbfb017c0d0821/d1f221f09f6bfffefb882c8f9e167483a.mp4"
  }
}
    
```

步骤3 调用**获取分段上传授权**接口，向OBS获取初始化上传任务的授权。

1. 选择**GET**请求方法并输入URI。
https://vod.cn-north-4.myhuaweicloud.com/v1.1/{project_id}/asset/authority?http_verb=POST&content_type=video/mp4&bucket=vod-bucket-65-cn-

```
north-4&object_key=05041fffa4002****f6dc009cc6f8f33/fea62a2845e43c37b4cbfb017c0d0821/d1f221f09f6bffe882c8f9e167483a.mp4
```

- http_verb: 分段上传时调用OBS接口的HTTP方法，具体操作需要的HTTP方法请参见[OBS API参考](#)。
- content_type: 文件类型对应的HTTP Content-type，您需要根据上传的文件类型进行设置。
 - 视频文件: video/*视频格式*，如：“video/mp4”。更多视频文件格式对应的请求头填写规则，请参见[表4-3](#)。
 - 音频文件: audio/*音频格式*，如：“audio/mp3”。更多音频文件格式对应的请求头填写规则，请参见[表4-4](#)。
 - 图片文件: image/*图片格式*，如：“image/png”。
 - 字幕文件: application/octet-stream。
- bucket: 存储视频文件的OBS桶，[步骤2.d](#)中返回的对应“bucket”字段。
- object_key: OBS桶中的对象名，[步骤2.d](#)中返回的对应“object”字段。

表 4-3 视频文件请求头填写说明

文件后缀	Content-Type
MP4	video/mp4
MOV	video/quicktime
MXF	application/mxf
TS	video/mp2t
MPG	video/mpeg
FLV	video/flv
WMV	video/x-ms-wmv
AVI	video/x-msvideo
M4V	video/m4v
F4V	application/f4v
MPEG	video/mpeg
M3U8	application/octet-stream
_3GP/3GP	video/3gpp
ASF	video/x-ms-asf
MKV	video/x-matroska
WEBM	video/webm
MPD	video/dash

表 4-4 音频文件请求头填写说明

文件后缀	Content-Type
MP3	audio/mp3
WMA	audio/wma
APE	audio/ape
FLAC	audio/flac
AAC	audio/aac
AC3	audio/ac3
MMF	audio/mmf
AMR	audio/amr
M4A	audio/m4a
M4R	audio/m4r
OGG	audio/ogg
WAV	audio/wav
WV	audio/wv
MP2	audio/mp2

2. 请求消息头中增加“X-Auth-Token”，值设置为步骤1中获取的用户Token。
3. 请求响应成功后，返回授权信息。

```
{
  "sign_str": "https://vod-bucket-65-cn-north-4.obs.cn-
north-4.myhuaweicloud.com:443/05041fffa4002****f6dc009cc6f8f33/
fea62a2845e43c37b4cbfb017c0d0821/d1f221f09f6bffe882c8f9e167483a.mp4?
uploads&AWSSignature=MZH0LUL329N1SSXNB3S4&Expires=1596706429&Signature=5D15iUcTko
WlpE2vn54FQZskXA%3D"
}
```

步骤4 调用OBS的[初始化上传段任务](#)接口，初始化视频分段上传任务。

须知

OBS不支持直接在API Explorer上验证，需要使用postman等第三方工具，构造请求做验证。

1. 选择POST请求方法并输入步骤3.c返回的已授权URI。
https://vod-bucket-65-cn-north-4.obs.cn-north-4.myhuaweicloud.com:443/05041fffa4002****f6dc009cc6f8f33/fea62a2845e43c37b4cbfb017c0d0821/d1f221f09f6bffe882c8f9e167483a.mp4?uploads&AWSSignature=MZH0LUL329N1SSXNB3S4&Expires=1596706429&Signature=5D15iUcTkoWlpE2vn54FQZskXA%3D
2. 请求消息头中增加“Content-Type”，值设置为文件类型，请根据实际上传。文件类型填写规则请参见步骤3。

3. 请求成功，返回初始化信息。

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<InitiateMultipartUploadResult xmlns="http://obs.cn-north-4.myhuaweicloud.com/doc/2015-06-30/">
  <Bucket>vod-bucket-65-cn-north-4</Bucket>
  <Key>05041ffa4002****f6dc009cc6f8f33/fea62a2845e43c37b4cbfb017c0d0821/
d1f221f09f6bffe882c8f9e167483a.mp4</Key>
  <UploadId>00000173C2ED862344C835374DFE33C8</UploadId>
</InitiateMultipartUploadResult>
```

步骤5 调用[获取分段上传授权](#)接口，获取分段上传的授权。

1. 选择GET请求方法并输入URI。

```
https://vod.cn-north-4.myhuaweicloud.com/v1.1/{project_id}/asset/authority?
http_verb=PUT&content_type=video/mp4&bucket=vod-bucket-65-cn-
north-4&object_key=05041ffa4002****f6dc009cc6f8f33/fea62a2845e43c37b4cbfb017c0d0821/
d1f221f09f6bffe882c8f9e167483a.mp4&content_md5=hHZXtgjYL2fmLpu%2byQ%2bXgg%3d
%3d&upload_id=00000173C2ED862344C835374DFE33C8&part_number=1
```

- http_verb: 分段上传的HTTP方法为PUT。
- content_type: 文件类型对应的HTTP Content-type，您需要根据上传的文件类型进行设置，文件类型填写规则请参见[步骤3](#)。
- bucket: 存储视频文件的OBS桶，[步骤2.d](#)中返回的对应“bucket”字段。
- object_key: OBS桶中的对象名，[步骤2.d](#)中返回的对应“object”字段。
- content_md5: 每个视频分段的MD5值，该值为必填项，具体生成方法可以参考[生成MD5值](#)。**注意**：如有特殊字符请求，需进行转义。
- upload_id: 上传任务的id，[步骤4.c](#)返回的“UploadId”字段。
- part_number: 每一个上传段的段号。

2. 请求消息头中增加“X-Auth-Token”，值设置为[步骤1](#)中获取的用户Token。

3. 请求响应成功后，返回分段上传的授权信息。

```
{
  "sign_str": "https://vod-bucket-65-cn-north-4.obs.cn-
north-4.myhuaweicloud.com:443/05041ffa4002****f6dc009cc6f8f33/
fea62a2845e43c37b4cbfb017c0d0821/d1f221f09f6bffe882c8f9e167483a.mp4?
AWSAccessKeyId=MZH0LUL329N1SSXNB3S4&Expires=1596708691&partNumber=1&uploadId=000001
73C2ED862344C835374DFE33C8&Signature=cjw3CmUFeNBFAuLWBTsPRp9NfsQ%3D"
}
```

步骤6 调用OBS的[上传段](#)接口，上传视频文件的第一个分段。

须知

OBS不支持直接在API Explorer上验证，需要使用postman等第三方工具，构造请求做验证。

1. 选择PUT请求方法并输入[步骤5.c](#)返回的已授权URI。

```
https://vod-bucket-65-cn-north-4.obs.cn-
north-4.myhuaweicloud.com:443/05041ffa4002****f6dc009cc6f8f33/
fea62a2845e43c37b4cbfb017c0d0821/d1f221f09f6bffe882c8f9e167483a.mp4?
AWSAccessKeyId=MZH0LUL329N1SSXNB3S4&Expires=1596708691&partNumber=1&uploadId=000001
73C2ED862344C835374DFE33C8&Signature=cjw3CmUFeNBFAuLWBTsPRp9NfsQ%3D
```

2. 请求消息头中增加“Content-Type”，值设置为“application/octet-stream”。若[步骤5.a](#)设置了“content_md5”参数，则请求消息头中必须增加“Content-MD5”，值设置为对应分段的MD5值（不需要转义），否则将会导致上传失败。

3. 请求Body中使用二进制方式上传视频文件的第一个分段。

4. 请求成功，返回状态码“200 OK”。

步骤7 参考操作**步骤5-步骤6**，上传视频文件的所有分段。

步骤8 调用**获取分段上传授权**接口，获取列举已上传段的授权。

1. 选择**GET**请求方法并输入URI。

```
https://vod.cn-north-4.myhuaweicloud.com/v1.1/{project_id}/asset/authority?
http_verb=GET&bucket=vod-bucket-65-cn-north-4&object_key=05041fffa4002****f6dc009cc6f8f33/
fea62a2845e43c37b4cbfb017c0d0821/
d1f221f09f6bffe882c8f9e167483a.mp4&upload_id=000001717E28524B44CEFF2C2CE1B06A
```

- http_verb: 列举已上传段的HTTP方法为GET。
- bucket: 存储视频文件的OBS桶，**步骤2.d**中返回的对应“bucket”字段。
- object_key: OBS桶中的对象名，**步骤2.d**中返回的对应“object”字段。
- upload_id: 上传任务的id，**步骤4.c**返回的“UploadId”字段。

2. 请求消息头中增加“X-Auth-Token”，值设置为**步骤1**中获取的用户Token。

3. 请求响应成功后，返回列表已上传段的授权信息。

```
{
  "sign_str": "https://vod-bucket-65-cn-north-4.obs.cn-
north-4.myhuaweicloud.com:443/05041fffa4002****f6dc009cc6f8f33/
fea62a2845e43c37b4cbfb017c0d0821/d1f221f09f6bffe882c8f9e167483a.mp4?
AWSAccessKeyId=MZH0LUL329N1SSXNB3S4&Expires=1596710054&uploadId=000001717E28524B44CE
FF2C2CE1B06A&Signature=G5EOUr488cyPlretp8lgQZmpPw0%3D"
}
```

步骤9 调用OBS接口**列举已上传的段**，获取视频文件所有上传段的相关信息。

须知

- OBS接口**列举已上传的段**支持响应段的取值范围[1,10000]，段大小[100KB,5GB]。默认最多响应1000段，如果用户分段数目超出1000，需参考OBS接口**列举已上传的段**进行处理。
- OBS不支持直接在API Explorer上验证，需要使用postman等第三方工具，构造请求做验证。

注意：请求消息头中的“Content-Type”属性设置成null或者不携带。

1. 选择**GET**请求方法并输入**步骤8.c**返回的已授权URI。

```
https://vod-bucket-65-cn-north-4.obs.cn-
north-4.myhuaweicloud.com:443/05041fffa4002****f6dc009cc6f8f33/
fea62a2845e43c37b4cbfb017c0d0821/d1f221f09f6bffe882c8f9e167483a.mp4?
AWSAccessKeyId=MZH0LUL329N1SSXNB3S4&Expires=1596710054&uploadId=000001717E28524B44CE
FF2C2CE1B06A&Signature=G5EOUr488cyPlretp8lgQZmpPw0%3D
```

2. 请求成功，返回已上传段的相关信息。

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<ListPartsResult xmlns="http://obs.cn-north-4.myhuaweicloud.com/doc/2015-06-30/">
  <Bucket>vod-bucket-65-cn-north-4</Bucket>
  <Key>05041fffa4002****f6dc009cc6f8f33/fea62a2845e43c37b4cbfb017c0d0821/
d1f221f09f6bffe882c8f9e167483a.mp4</Key>
  <UploadId>00000173C2ED862344C835374DFE33C8</UploadId>
  <Initiator>
    <ID>d9235b9f3cf549499924f6de095241af:a96ecebcb3994e34b074e48f3dfc8237</ID>
    <DisplayName>xxx</DisplayName>
  </Initiator>
  <Owner>
    <ID>d9235b9f3cf549499924f6de095241af</ID>
    <DisplayName>xxx</DisplayName>
  </Owner>
  <StorageClass>STANDARD</StorageClass>
  <PartNumberMarker>0</PartNumberMarker>
  <NextPartNumberMarker>4</NextPartNumberMarker>
```

```
<MaxParts>1000</MaxParts>
<IsTruncated>>false</IsTruncated>
<Part>
  <PartNumber>1</PartNumber>
  <LastModified>2020-08-06T09:05:10.192Z</LastModified>
  <ETag>"847657b608d82f67e62e9bbec90f9782"</ETag>
  <Size>10000000</Size>
</Part>
<Part>
  <PartNumber>2</PartNumber>
  <LastModified>2020-08-06T08:02:02.821Z</LastModified>
  <ETag>"9a6a36ed9086a3a2fea130220e1e809c"</ETag>
  <Size>10000000</Size>
</Part>
<Part>
  <PartNumber>3</PartNumber>
  <LastModified>2020-08-06T08:04:19.711Z</LastModified>
  <ETag>"3a3afe552832bee0faf081c1e720067e"</ETag>
  <Size>10000000</Size>
</Part>
<Part>
  <PartNumber>4</PartNumber>
  <LastModified>2020-08-06T07:23:17.160Z</LastModified>
  <ETag>"6335af859c20ccef26b27ea691e7ecf7"</ETag>
  <Size>7472445</Size>
</Part>
</ListPartsResult>
```

步骤10 调用[获取分段上传授权](#)接口，获取合并段授权。

1. 选择**GET**请求方法并输入URI。

```
https://vod.cn-north-4.myhuaweicloud.com/v1.1/{project_id}/asset/authority?
http_verb=POST&bucket=vod-bucket-65-cn-north-4&object_key=05041fffa4002****f6dc009cc6f8f33/
fea62a2845e43c37b4cbfb017c0d0821/d1f221f09f6bffe882c8f9e167483a.mp4&upload_id=00000173C2ED862344C835374DFE33C8
```

- http_verb: 合并段的HTTP方法为POST。
- bucket: 存储视频文件的OBS桶，[步骤2.d](#)中返回的对应“bucket”字段。
- object_key: OBS桶中的对象名，[步骤2.d](#)中返回的对应“object”字段。
- upload_id: 上传任务的id，[步骤4.c](#)返回的“UploadId”字段。

2. 请求消息头中增加“X-Auth-Token”，值设置为[步骤1](#)中获取的用户Token。

3. 请求响应成功后，返回合上传段的授权信息。

```
{
  "sign_str": "https://vod-bucket-65-cn-north-4.obs.cn-
north-4.myhuaweicloud.com:443/05041fffa4002****f6dc009cc6f8f33/
fea62a2845e43c37b4cbfb017c0d0821/d1f221f09f6bffe882c8f9e167483a.mp4?
AWSAccessKeyId=MZH0LUL329N1SSXNB3S4&Expires=1596709340&uploadId=000001717E28524B44CE
FF2C2CE1B06A&Signature=Pa6laMbH1Ofa0Vi%2BCbkdgWzm70%3D"
}
```

步骤11 调用OBS的[合并段](#)接口，将上传的分段合并成一个视频文件。

须知

OBS不支持直接在API Explorer上验证，需要使用postman等第三方工具，构造请求做验证。

1. 选择**POST**请求方法并输入[步骤10.c](#)返回的已授权URI。

```
https://vod-bucket-65-cn-north-4.obs.cn-
north-4.myhuaweicloud.com:443/05041fffa4002****f6dc009cc6f8f33/
fea62a2845e43c37b4cbfb017c0d0821/d1f221f09f6bffe882c8f9e167483a.mp4?
AWSAccessKeyId=MZH0LUL329N1SSXNB3S4&Expires=1596708691&partNumber=1&uploadId=000001
73C2ED862344C835374DFE33C8&Signature=cjw3CmUFeNBFAuLWBTsPRp9NfsQ%3D
```

2. 请求消息头中增加“Content-Type”，值设置为“application/xml”。
3. Request Body中传入参数如下：

```
<CompleteMultipartUpload>
  <Part>
    <PartNumber>1</PartNumber>
    <ETag>"847657b608d82f67e62e9bbec90f9782"</ETag>
  </Part>
  <Part>
    <PartNumber>2</PartNumber>
    <ETag>"9a6a36ed9086a3a2fea130220e1e809c"</ETag>
  </Part>
  <Part>
    <PartNumber>3</PartNumber>
    <ETag>"3a3afe552832bee0faf081c1e720067e"</ETag>
  </Part>
  <Part>
    <PartNumber>4</PartNumber>
    <ETag>"6335af859c20cccf26b27ea691e7ecf7"</ETag>
  </Part>
</CompleteMultipartUpload>
```

- PartNumber: 上传的段号。
 - ETag: 上传段成功后返回的ETag值，[步骤9.b](#)中的返回信息有每个段对应的ETag值。
4. 请求成功，返回合并段的相关信息。

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<CompleteMultipartUploadResult xmlns="http://obs.cn-north-4.myhuaweicloud.com/doc/2015-06-30/">
  <Location>05041fffa4002****f6dc009cc6f8f33/fea62a2845e43c37b4cbfb017c0d0821/d1f221f09f6bffe6b882c8f9e167483a.mp4</Location>
  <Bucket>vod-bucket-65-cn-north-4</Bucket>
  <Key>05041fffa4002****f6dc009cc6f8f33/fea62a2845e43c37b4cbfb017c0d0821/d1f221f09f6bffe6b882c8f9e167483a.mp4</Key>
  <ETag>"87f450dd18b666dfcdf902ac6b162b5a-4"</ETag>
</CompleteMultipartUploadResult>
```

步骤12 调用[确认媒资上传](#)接口，完成媒资上传确认。

1. 选择POST请求方法并输入URI。
https://vod.cn-north-4.myhuaweicloud.com/v1.0/{project_id}/asset/status/uploaded
2. 请求消息头中增加“X-Auth-Token”，值设置为[步骤1](#)中获取的用户Token。
3. Request Body中传入参数如下：

```
{
  "asset_id": "fea62a2845e43c37b4cbfb017c0d0821",
  "status": "CREATED"
}
```

4. 请求响应成功后，返回媒资ID。

```
{
  "asset_id": "fea62a2845e43c37b4cbfb017c0d0821"
}
```

----结束

分段上传 Java 语言代码示例

获取分段上传Java语言的示例代码，如下所示：

- [Java](#)
- [POM文件](#)

分段上传 JavaScript 语言代码示例

获取分段上传JavaScript语言的示例代码，如[JavaScript](#)所示。

分段上传 Python 语言代码示例

获取分段上传Python语言的示例代码，如[Python](#)所示。

分段上传 Go 语言代码示例

获取分段上传Go语言的示例代码，如[Go](#)所示。

分段上传 PHP 语言代码示例

获取分段上传PHP语言的示例代码，如[PHP](#)所示。

4.3 示例 3：获取媒资详情

场景描述

若您需要查询一个或多个媒资的基本信息、转码文件生成信息、截图信息及审核信息，您可以调用VOD API进行查询。

本章节指导用户如何使用API调用的方式获取存储在VOD中的媒资的详情信息。

前提条件

- 已明确需要查询的媒资所在区域信息，详见[使用前必读](#)。
- 已获取需要查询的媒资所在区域的项目ID，具体获取方法请参见[获取项目ID](#)。

总体流程

1. [获取用户Token](#)
2. [查询媒资详细信息](#)

操作步骤

步骤1 获取用户Token，用于调用点播接口时进行认证鉴权。

详细信息请参见[构造请求](#)。以“华北-北京四”为示例，若您需要调用其它区域的点播接口，请替换成对应区域的[IAM终端节点](#)。

```
POST https://iam.cn-north-4.myhuaweicloud.com/v3/auth/tokens
Content-Type: application/json
```

```
{
  "auth": {
    "identity": {
      "methods": [
        "password"
      ],
      "password": {
        "user": {
          "name": "username",
          "password": "password",
          "domain": {
```

```

    "name": "domainname"
  }
}
},
"scope": {
  "project": {
    "name": "projectname"
  }
}
}
}
}

```

如图4-5所示，红框中的信息为获取用户Token。

图 4-5 获取用户 Token

```

cache-control → no-cache, no-store, must-revalidate
connection → keep-alive
content-length → 15185
content-type → application/json; charset=UTF-8
date → Wed, 29 May 2019 08:02:38 GMT
expires → Thu, 01 Jan 1970 00:00:00 GMT
pragma → no-cache
server → api-gateway
strict-transport-security → max-age=31536000; includeSubdomains;
via → proxy A
x-content-type-options → nosniff
x-download-options → noopen
x-frame-options → SAMEORIGIN
x-lam-trace-id → token_global_null_2fed19b42067d0b94ae73dbe0b8b7841
x-request-id → fd8e4d01ae9caf958af20cbdb2c88e2
x-subject-token → MElYdWVkaZzBhcNAQ:CellYdCCGGQCAQJExDTAlBglghkgBQzQMfAgEwghbFBgkqhkiG9w0BBwGggha2BllWnsidG9-ZW40OnsiZXhwaXk:19hdCl6jwMTkMUDUHMcBUMDg6MDJ6MzguNTgxMDAwWlIdm1IdGhvZHMlOlscGFzc3dcmQz5swY2F0YWwzJzwn-yN0WpSwyRGlY3upPFOauVL-E275QpmEQPG4qGpVdYeo2Hw8ZDFpt86AZt-RGduoWLe95negpR0PGKzgbuRiWYAtdQL3xGxVvC1Um8XlADmwO2TWgibmzPwF8vP3F20TK4v9XhwWHUxH2fo+KScgsXHAzv6VURCIWMDV+UeE5wZo7v82C77p289d8lly63oQzZmTyrIaupZl6U4PK2J+rW4pIwEszdvrRQLLYOzxinGG3Af
x-xss-protection → 1; mode=block

```

步骤2 调用查询媒资信息接口，查询媒资详情信息。

1. 选择GET请求方法并输入URI。

```

https://{endpoint}/v1.0/{project_id}/asset/details?
asset_id=41cff636d5b12a51e7eb2838bbf45201&categories=base_info,transcode_info&thumbnail_info,r
eview_info

```

“categories”为查询媒资信息的类型，您可以选择一个或者多个类型进行查询，具体取值如下所示：

- base_info: 查询媒资基本信息，包含媒资标题、分类、描述、文件类型、原始文件地址等。
- transcode_info: 查询媒资转码文件信息，包含使用的转码模板、转码后的媒资信息等。
- thumbnail_info: 查询截图信息，包含截图状态、截图存储地址等。
- review_info: 审核信息，包含审核是否通过，文本审核结果、封面审核结果和视频审核结果等。

📖 说明

仅转码、截图或审核成功时，才可查询到对应的转码信息、截图信息和审核信息。

2. 请求消息头中增加“X-Auth-Token”，值设置为步骤1中获取的用户Token。

3. 请求响应成功后，返回媒资详情信息。

```

{
  "asset_info_array": [ {
    "asset_id": "652c1e4085afeb22fdc256c6757d751b",

```

```

"status": "PUBLISHED",
"description": "Asset meta is published",
"base_info": {
  "title": "okFLV.flv",
  "video_name": "okFLV.flv",
  "description": null,
  "category_id": -1,
  "category_name": "其他",
  "create_time": "20190704144303",
  "last_modified": "20190704144303",
  "video_type": "FLV",
  "tags": null,
  "meta_data": {
    "pack_type": null,
    "codec": "H.264",
    "duration": 244,
    "video_size": 13682041,
    "width": 512,
    "height": 288,
    "bit_rate": 448,
    "frame_rate": 30,
    "quality": null
  },
  "video_url": "https://651.cdn-vod.huaweicloud.com/asset/652c1e4085afeb22fdc256c6757d751b/45c2493abe68de3dac7e98b0dadcf8ce.flv",
  "sign_url": "",
  "cover_info_array": [ {
    "cover_url": "https://651.cdn-vod.huaweicloud.com/asset/652c1e4085afeb22fdc256c6757d751b/cover/Cover0.jpg"
  } ],
  "subtitle_info": [ ],
  "source_path": {
    "bucket": "obs-host",
    "location": "cn-north-4",
    "object": "1/okFLV.flv"
  },
  "output_path": {
    "bucket": "obs-host",
    "location": "cn-north-4",
    "object": "output/652c1e4085afeb22fdc256c6757d751b/"
  }
},
"play_info_array": [ {
  "play_type": "hls",
  "url": "https://651.cdn-vod.huaweicloud.com/asset/652c1e4085afeb22fdc256c6757d751b/play_video/index.m3u8",
  "encrypted": 0,
  "meta_data": {
    "pack_type": null,
    "codec": "H.264",
    "duration": 0,
    "video_size": 0,
    "width": 0,
    "height": 0,
    "bit_rate": 0,
    "frame_rate": 0,
    "quality": null
  }
}, {
  "play_type": "hls",
  "url": "https://651.cdn-vod.huaweicloud.com/asset/652c1e4085afeb22fdc256c6757d751b/play_video/b5d498082bbcff7a2055041d803ae2f9_1.m3u8",
  "encrypted": 0,
  "meta_data": {
    "pack_type": null,
    "codec": "H.264",
    "duration": 205,
    "video_size": 14669824,
    "width": 512,

```

```

    "hight" : 288,
    "bit_rate" : 534,
    "frame_rate" : 0,
    "quality" : null
  }
}
}],
"is_truncated" : 0,
"total" : 1
}
{
  "asset_info_array" : [ {
    "asset_id" : "652c1e4085afeb22fdc256c6757d751b",
    "status" : "PUBLISHED",
    "description" : "Asset meta is published",
    "base_info" : {
      "title" : "okFLV.flv",
      "video_name" : "okFLV.flv",
      "description" : null,
      "category_id" : -1,
      "category_name" : "其他",
      "create_time" : "20190704144303",
      "last_modified" : "20190704144303",
      "video_type" : "FLV",
      "tags" : null,
      "meta_data" : {
        "pack_type" : null,
        "codec" : "H.264",
        "duration" : 244,
        "video_size" : 13682041,
        "width" : 512,
        "hight" : 288,
        "bit_rate" : 448,
        "frame_rate" : 30,
        "quality" : null
      },
      "video_url" : "https://651.cdn-vod.huaweicloud.com/asset/652c1e4085afeb22fdc256c6757d751b/45c2493abe68de3dac7e98b0dadcf8ce.flv",
      "sign_url" : "",
      "cover_info_array" : [ {
        "cover_url" : "https://651.cdn-vod.huaweicloud.com/asset/652c1e4085afeb22fdc256c6757d751b/cover/Cover0.jpg"
      } ],
      "subtitle_info" : [ ],
      "source_path" : {
        "bucket" : "obs-host",
        "location" : "cn-north-4",
        "object" : "1/okFLV.flv"
      },
      "output_path" : {
        "bucket" : "obs-host",
        "location" : "cn-north-4",
        "object" : "output/652c1e4085afeb22fdc256c6757d751b/"
      }
    },
    "play_info_array" : [ {
      "play_type" : "hls",
      "url" : "https://651.cdn-vod.huaweicloud.com/asset/652c1e4085afeb22fdc256c6757d751b/play_video/index.m3u8",
      "encrypted" : 0,
      "meta_data" : {
        "pack_type" : null,
        "codec" : "H.264",
        "duration" : 0,
        "video_size" : 0,
        "width" : 0,
        "hight" : 0,
        "bit_rate" : 0,
        "frame_rate" : 0,

```

```

    "quality" : null
  }
}, {
  "play_type" : "hls",
  "url" : "https://651.cdn-vod.huaweicloud.com/asset/652c1e4085afeb22fdc256c6757d751b/
play_video/b5d498082bbcff7a2055041d803ae2f9_1.m3u8",
  "encrypted" : 0,
  "meta_data" : {
    "pack_type" : null,
    "codec" : "H.264",
    "duration" : 205,
    "video_size" : 14669824,
    "width" : 512,
    "hight" : 288,
    "bit_rate" : 534,
    "frame_rate" : 0,
    "quality" : null
  }
}]
}],
"is_truncated" : 0,
"total" : 1
}

```

----结束

4.4 示例 4：视频处理

场景描述

音视频上传时，可以设置媒资处理参数，上传完成后，自动根据设置的参数进行音视频处理。若您需要对已上传的视频进行处理，您也可以调用VOD API实现。

本章节指导用户如何使用API调用的方式处理已上传的视频文件，包括视频转码、截图、加密等，支持同时启动多种处理操作，也可以只启动一种处理操作。

前提条件

- 已明确需要处理的视频文件所在区域信息，详见[使用前必读](#)。
- 已获取需要处理的视频文件所在区域的项目ID，具体获取方法请参见[获取项目ID](#)。

总体流程

1. [获取用户Token](#)
2. [（可选）创建转码模板](#)
3. [（可选）设置搭建的密钥管理服务的URL](#)
4. [创建视频处理任务](#)
5. [查询视频处理执行情况](#)

操作步骤

步骤1 获取用户Token，用于调用点播接口时进行认证鉴权。

详细信息请参见[构造请求](#)。以“华北-北京四”为示例，若您需要调用其它区域的点播接口，请替换成对应区域的[IAM终端节点](#)。

POST https://iam.cn-north-4.myhuaweicloud.com/v3/auth/tokens
Content-Type: application/json

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

如图4-6所示，红框中的信息为获取用户Token。

图 4-6 获取用户 Token



步骤2 在视频点播控制台创建自定义转码模板，具体操作请参见[转码设置](#)。

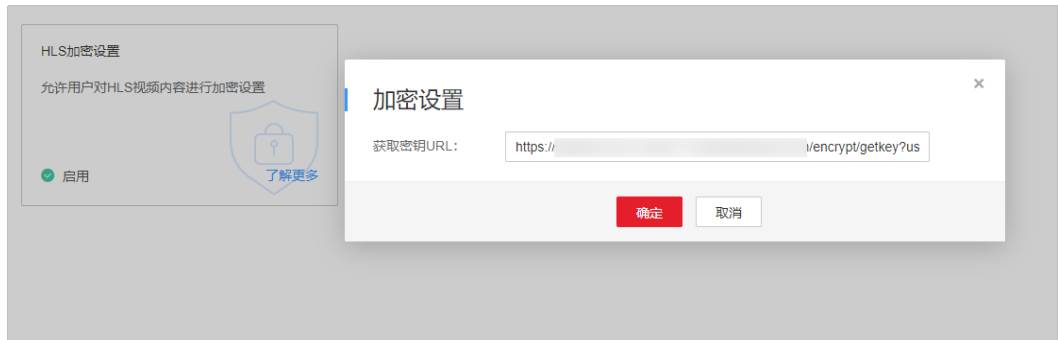
您也可以使用点播服务提供的系统转码模板。创建转码模板后，您可以在视频点播控制台获取需要使用的转码模板的模板名称。

图 4-7 获取转码模板名称

模板组名称	类型	描述	操作
non_transcoding_template_group	系统模板组	不转码模板	
original_template_group	系统模板组	普通转码模板，只转换格式，不改变分辨率和码率	编辑
system_template_group 默认	系统模板组	视频转码模板，不可修改分辨率和码率	编辑
adaptive_template_group	系统模板组	视频自适应转码模板，可修改分辨率和码率	编辑
test	自定义模板组		编辑 删除

- 步骤3** 在视频点播控制台配置搭建的密钥管理服务的URL，具体操作请参见[HLS加密设置](#)。仅能对转码输出格式为HLS的视频进行加密。若不需要进行加密操作，可以跳过该步骤。

图 4-8 加密设置



- 步骤4** 创建视频处理任务。

1. 选择**POST**请求方法并输入URI。
`https://{endpoint}/v1.0/{project_id}/asset/process`
2. 请求消息头中增加“X-Auth-Token”，值设置为**步骤1**中获取的用户Token。
3. Request Body中传入参数如下：

```
{
  "asset_id": "b4f39691d66cc0ef75d62ee567146e11",
  "template_group_name": "test",
  "auto_encrypt": 0,
  "thumbnail": {
    "type": "time",
    "time": 12
  },
  "subtitle_id": [1]
}
```

📖 说明

仅在**示例1：媒资上传（20M以下）**时，请求参数设置了“subtitles”时，才可以在视频处理时，更换字幕文件，设置的“subtitle_id”才有效。

4. 请求响应成功后，返回处理媒资ID。

```
{
  "asset_id": "b4f39691d66cc0ef75d62ee567146e11"
}
```

- 步骤5** 调用查询详细信息接口查询视频处理执行情况，具体调用方法请参见[示例3：获取媒资详情](#)。

视频处理任务执行的时间与视频文件的大小、设置的处理操作个数等有关，建议视频任务提交5-10分钟后再查询执行情况。

----结束

5 媒资上传

5.1 创建媒资：上传方式

功能介绍

调用该接口创建媒资时，需要将对应的媒资文件上传到点播服务的OBS桶中。若上传的单媒资文件大小小于20M，则可以直接用PUT方法对该接口返回的地址进行上传。具体使用方法请参考[示例1：媒资上传（20M以下）](#)。若上传的单个媒资大小大于20M，则需要进行二进制流分割后上传，该接口的具体使用方法请参考[示例2：媒资分段上传（20M以上）](#)。

URI

POST /v1.0/{project_id}/asset

表 5-1 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

请求参数

表 5-2 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

参数	是否必选	参数类型	描述
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

表 5-3 请求 Body 参数

参数	是否必选	参数类型	描述
title	是	String	媒资标题，长度不超过128个字节，UTF-8编码。
description	否	String	视频描述，长度不超过1024个字节。
video_name	是	String	音视频文件名，长度不超过128个字节。 文件名后缀可选。
video_type	是	String	上传音视频文件的格式。 取值如下： <ul style="list-style-type: none"> • 视频文件：MP4、TS、MOV、MXF、MPG、FLV、WMV、AVI、M4V、F4V、MPEG、3GP、ASF、MKV、WEBM • 音频文件：MP3、OGG、WAV、WMA、APE、FLAC、AAC、AC3、MMF、AMR、M4A、M4R、WV、MP2 若上传格式为音频文件，则不支持转码、添加水印和字幕。
category_id	否	Integer	媒资分类ID。 您可以调用 创建媒资分类 接口或在点播控制台的分类设置中创建对应的媒资分类，并获取分类ID。 说明 若不设置或者设置为-1，则上传的音视频归类到系统预置的“其它”分类中。
video_md5	否	String	建议参考《API参考》附录下生成MD5值的媒资上传和更新示例。

参数	是否必选	参数类型	描述
cover_type	否	String	封面图片文件类型。 取值如下： <ul style="list-style-type: none"> • JPG • PNG 上传后的封面名称是固定的，后缀名为封面类型缩写。例如 cover0.jpg, cover1.png 若不指定类型，则封面文件无后缀名。 说明 如果设置了图片格式，则不会执行首帧截图作为封面动作，需自行上传封面。
cover_md5	否	String	封面图片文件MD5值。建议参考《API参考》附录下生成MD5值的媒资上传和更新示例。
subtitles	否	Array of Subtitle objects	字幕文件信息
tags	否	String	视频标签。 单个标签不超过24个字节，最多不超过16个标签。 多个用逗号分隔，UTF8编码。
auto_publish	否	Integer	是否自动发布。 取值如下： <ul style="list-style-type: none"> • 0：表示不自动发布。 • 1：表示自动发布。 默认值：1。
template_group_name	否	String	转码模板组名称。 若不为空，则使用指定的转码模板对上传的音视频进行转码，您可以在视频点播控制台配置转码模板，具体请参见《点播服务用户指南》中的“转码设置”。 说明 若同时设置了“ template_group_name ”和“ workflow_name ”字段，则“ template_group_name ”字段生效。

参数	是否必选	参数类型	描述
auto_encrypt	否	Integer	是否自动加密。 取值如下： <ul style="list-style-type: none"> 0: 表示不加密。 1: 表示需要加密。 默认值: 0。 加密与转码必须要一起进行，当需要加密时，转码参数不能为空，且转码输出格式必须要为HLS。
auto_preheat	否	String	是否自动预热到CDN。 取值如下： <ul style="list-style-type: none"> 0: 表示不自动预热。 1: 表示自动预热。 默认值: 0。
thumbnail	否	Thumbnail object	截图参数
review	否	Review object	媒资审核参数。 说明 仅“亚太-新加坡”点播服务支持此功能。
workflow_name	否	String	工作流名称。若不为空，则使用指定的工作流对上传的音视频进行处理，您可以在视频点播控制台配置工作流，具体请参见《点播服务用户指南》中的“工作流设置”。

表 5-4 Subtitle

参数	是否必选	参数类型	描述
id	是	Integer	字幕id。 取值范围: [1,16]。
type	是	String	字幕文件类型，目前暂只支持“SRT”和“VTT”。
language	是	String	字幕语言类型。
name	否	String	字幕文件名称。
md5	否	String	字幕文件的MD5值。
description	否	String	字幕描述。

表 5-5 Thumbnail

参数	是否必选	参数类型	描述
type	是	String	截图类型。 取值如下： <ul style="list-style-type: none"> time: 每次进行截图的间隔时间。 dots: 按照指定的时间点截图。 quantity: 按照指定张数，根据视频时长等分视频截图。
quantity	否	Integer	type 取值为quantity时必填。 按照指定张数，根据视频时长等分视频截图。 取值范围：[1,10]之间的整数。
quantity_time	否	Integer	type 取值为quantity时选填。 按照指定时间间隔取指定张数截图。 取值范围：[0,2147483647]之间的整数。
time	否	Integer	根据时间间隔采样时的时间间隔值。单位：秒。 type 取值为time时。 默认值：12 取值范围：[0,100]之间的整数。
dots	否	Array of integers	type 取值为dots时必填。指定时间截图时的时间点数组。
cover_position	否	Integer	该值表示指定第几张截图作为封面。 默认值：1。
format	否	Integer	截图文件格式。 取值如下： <ul style="list-style-type: none"> 1: jpg。 默认值：1。

参数	是否必选	参数类型	描述
aspect_ratio	否	Integer	纵横比，图像缩放方式。 取值如下： <ul style="list-style-type: none"> 0: 自适应（保持原有宽高比）。 1: 16:9。 默认值：0。
max_length	否	Integer	截图最长边的尺寸。 单位：像素。 宽边尺寸按照该尺寸与原始视频像素等比缩放计算。 默认值：480。

表 5-6 Review

参数	是否必选	参数类型	描述
template_id	否	String	审核模板ID。您可以在视频点播控制台配置审核模板后获取，具体请参见《点播服务用户指南》中的“审核设置”。 说明 仅“亚太-新加坡”点播服务支持此功能。
interval	否	Integer	截图检测时间间隔，取值范围为(0,100]，该参数在请求参数中忽略。
politics	否	Integer	鉴政内容检测置信度，取值范围为-1或者[0,100]，该参数在请求参数中忽略。 置信度越高，说明审核结果越可信。未开启或设置为0时，表示未进行此项检测。politics、terrorism、porn三个参数不能同时为0。
terrorism	否	Integer	鉴恐内容的检测置信度，取值范围为-1或者[0,100]，该参数在请求参数中忽略。 置信度越高，说明审核结果越可信。未开启或设置为0时，表示未进行此项检测。politics、terrorism、porn三个参数不能同时为0。

参数	是否必选	参数类型	描述
porn	否	Integer	鉴黄内容的检测置信度，取值范围为-1或者[0,100]，该参数在请求参数中忽略。 置信度越高，说明审核结果越可信。未开启或设置为0时，表示未进行此项检测。politics、terrorism、porn三个参数不能同时为0。

响应参数

状态码：200

表 5-7 响应 Body 参数

参数	参数类型	描述
asset_id	String	媒体ID
video_upload_url	String	视频上传URL
cover_upload_url	String	封面上传地址
subtitle_upload_urls	Array of strings	字幕文件上传url数组
target	File_addr object	媒资存储参数信息。

表 5-8 File_addr

参数	参数类型	描述
bucket	String	OBS的bucket名称。
location	String	桶所在的区域名，如“华北-北京四”的区域名为“cn-north-4”，创建的桶所在区域必须和点播服务所在区域保持一致。
object	String	文件的存储路径。

状态码：403

表 5-9 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

创建媒资

POST https://{endpoint}/v1/{project_id}/asset

Content-Type: application/json

```
{
  "title": "Avatar test test",
  "description": "Avatar, test",
  "category_id": -1,
  "tags": "mytags",
  "video_name": "Avatar_480P.mp4",
  "video_type": "MP4",
  "video_md5": "a945d4b3d8fc317190a9332fe856f03d",
  "cover_type": "JPG",
  "cover_md5": "a655d4b3d8fc758691a9332fe387f26c",
  "auto_publish": 0,
  "subtitles": [
    {
      "id": 1,
      "language": "CN",
      "type": "SRT",
      "md5": "SqcyFjJZoDZaP8oKIY6rgQ==",
      "description": "AAAAA"
    }
  ]
}
```

响应示例

状态码：200

处理成功返回。

```
{
  "asset_id": "f488337c31c8e4622f1590735b134c65",
  "video_upload_url": "https://obs.cn-north-4.myhuaweicloud.com:443/obs-vod-1/%7Bproject_id%7D/f488337c31c8e4622f1590735b134c65/Avatar_480P.mp4?AWSAccessKeyId=CBN2J*****0RCSN&Expires=1518147618&Signature=kZYh0hEos2V*****AHGyXA%3D",
  "cover_upload_url": "https://obs.cn-north-4.myhuaweicloud.com:443/obs-vod-1/%7Bproject_id%7D/f488337c31c8e4622f1590735b134c65/cover/Cover0.jpg?AWSAccessKeyId=CBN2J*****0RCSN&Expires=1518147619&Signature=kZYh0hEos2V*****AHGyXA%3D",
  "subtitle_upload_urls": [ "https://obs-vod-1.obs-cn-north-4.myhuaweicloud.com:443/14ce1d4437164aba8b364ce15866154e/53a018d2dc53ca07eb5a07a839205c9d/subtitle/1.srt?AWSAccessKeyId=CBN2J*****0RCSN&Expires=1534760131&Signature=kZYh0hEos2V*****AHGyXA%3D" ],
  "target": {
    "bucket": "obs-vod-1",
    "location": "cn-north-4",
    "object": "093bb6b6c4fc460ab90a40d8b821dda3/a2053aef99725711dad3e02dc6cd5f89/0a9b70035b78b8a19c6d9e7c2693d93c.mp4"
  }
}
```

```
}  
}
```

状态码：403

处理失败返回。

```
{  
  "error_code" : "VOD.10064",  
  "error_msg" : "Media asset classification does not exist, please check."  
}
```

状态码

状态码	描述
200	处理成功返回。
403	处理失败返回。

错误码

请参见[错误码](#)。

5.2 获取分段上传授权

功能介绍

客户端请求创建媒资时，如果媒资文件超过20MB，需采用分段的方式向OBS上传，在每次与OBS交互前，客户端需通过此接口获取到授权方可与OBS交互。

该接口可以获取初始化多段上传任务、上传段、合并段、列举已上传段、取消段合并的带有临时授权的URL，用户需要根据OBS的接口文档配置相应请求的HTTP请求方法、请求头、请求体，然后请求对应的带有临时授权的URL。

视频分段上传方式和OBS的接口文档保持一致，包括HTTP请求方法、请求头、请求体等各种入参，此接口的作用是为用户生成带有鉴权信息的URL（鉴权信息即sign_str），用来替换OBS接口中对应的URL，临时给用户开通向点播服务的桶上传文件的权限。

调用获取授权接口时需要传入bucket、object_key、http_verb，其中bucket和object_key是由[创建媒资：上传方式](#)接口中返回的响应体中的target字段获得的bucket和object，http_verb需要根据指定的操作选择。

URI

GET /v1.1/{project_id}/asset/authority

表 5-10 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

表 5-11 Query 参数

参数	是否必选	参数类型	描述
http_verb	是	String	分段上传时调用OBS接口的HTTP方法，具体操作需要的HTTP方法请参考OBS的接口文档。 <ul style="list-style-type: none"> 初始化上传任务：POST 上传段：PUT 合并段：POST 取消段：DELETE 列举已上传段：GET
bucket	是	String	桶名。 调用 创建媒资：上传方式 接口中返回的响应体中的target字段获得的bucket值。
object_key	是	String	对象名。 调用 创建媒资：上传方式 接口中返回的响应体中的target字段获得的object值。
content_type	否	String	文件类型对应的content-type，当为上传任务初始化场景该字段必填。 参数填写可参考 媒资分段上传（20M以上） 。 <ul style="list-style-type: none"> 视频文件：video/视频格式，如：“video/mp4” 音频文件：audio/音频格式，如：“audio/mp3” 图片文件：image/图片格式，如：“image/png” 字幕文件：application/octet-stream
content_md5	否	String	上传段时每段的MD5。

参数	是否必选	参数类型	描述
upload_id	否	String	每一个上传任务的id, 是OBS初始化视频分段上传任务后返回的值。除上传任务初始化场景外, 该字段必填。
part_number	否	Integer	上传段时每一段的id。 取值范围: [1,10000]。

请求参数

表 5-12 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token, 使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取 (响应消息头中 X-Subject-Token的值) 。
Authorization	否	String	使用AK/SK方式认证时必选, 携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选, 请求的发生时间。

响应参数

状态码: 200

表 5-13 响应 Body 参数

参数	参数类型	描述
sign_str	String	带授权签名字符串的URL。具体调用示例请参见 示例2: 媒资分段上传 (20M以上) 。示例: https://{obs_domain}/{bucket}?AWSAccessKeyId={AccessKeyId}&Expires={ExpiresValue}&Signature={Signature}

状态码: 403

表 5-14 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

- 初始化上传任务：**
 GET https://{endpoint}/v1.1/{project_id}/asset/authority?
 http_verb=POST&content_type={type}&bucket={bucket}&object_key={objectKey}
- 上传段：**
 GET https://{endpoint}/v1.1/{project_id}/asset/authority?
 http_verb=PUT&content_md5={md5}&part_number={num}&upload_id={id}&bucket={bucket}&object_key={objectKey}
- 合并段：**
 GET https://{endpoint}/v1.1/{project_id}/asset/authority?
 http_verb=POST&upload_id={id}&bucket={bucket}&object_key={objectKey}
- 取消段：**
 GET https://{endpoint}/v1.1/{project_id}/asset/authority?
 http_verb=DELETE&bucket={bucket}&object_key={objectKey}&upload_id={uploadId}
- 列举已上传的段：**
 GET https://{endpoint}/v1.1/{project_id}/asset/authority?
 http_verb=GET&bucket={bucket}&object_key={objectKey}&upload_id={uploadId}

响应示例

状态码：200

处理成功返回。

```
{
  "sign_str" : "https://obs.cn-north-4.myhuaweicloud.com:443/obs-vod-1/%7Bproject_id%7D/f488337c31c8e4622f1590735b134c65/Avatar_480P.mp4?AWSAccessKeyId=CBN2*****ORCSN&Expires=1518147618&Signature=kZYh0hEos2V*****AHGyXA%3D"
}
```

状态码：403

处理失败返回。

```
{
  "error_code" : "VOD.10053",
  "error_msg" : "The request parameter is illegal, illegal field: {xx}."
}
```

状态码

状态码	描述
200	处理成功返回。
403	处理失败返回。

错误码

请参见[错误码](#)。

5.3 确认媒资上传

功能介绍

媒资分段上传完成后，需要调用此接口通知点播服务媒资上传的状态，表示媒资上传创建完成。

URI

POST /v1.0/{project_id}/asset/status/uploaded

表 5-15 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

请求参数

表 5-16 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

表 5-17 请求 Body 参数

参数	是否必选	参数类型	描述
asset_id	是	String	VOD分配的媒资ID，仅支持查询，不支持修改。

参数	是否必选	参数类型	描述
status	是	String	上传状态。 取值如下： <ul style="list-style-type: none"> • CREATED：创建成功。 • FAILED：创建失败。 • CANCELLED：创建取消。

响应参数

状态码：200

表 5-18 响应 Body 参数

参数	参数类型	描述
asset_id	String	VOD分配的媒资ID，仅支持查询，不支持修改。

状态码：403

表 5-19 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

通知点播服务媒资上传的状态

```
POST https://{endpoint}/v1.0/{project_id}/asset/status/uploaded
```

```
Content-Type: application/json
```

```
{
  "asset_id": "f488337c31c8e4622f1590735b134c65",
  "status": "CREATED"
}
```

响应示例

状态码：200

处理成功返回。

```
{
  "asset_id": "f488337c31c8e4622f1590735b134c65"
}
```


状态码：403

处理失败返回。

```
{
  "error_code": "VOD.10062",
  "error_msg": "Media asset or resource does not exist, please check."
}
```

状态码

状态码	描述
200	处理成功返回。
403	处理失败返回。

错误码

请参见[错误码](#)。

5.4 桶授权

功能介绍

用户可以通过该接口将OBS桶授权给点播服务或取消点播服务的授权。

URI

PUT /v1.0/{project_id}/asset/authority

表 5-20 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

请求参数

表 5-21 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

参数	是否必选	参数类型	描述
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

表 5-22 请求 Body 参数

参数	是否必选	参数类型	描述
bucket	是	String	OBS桶名称。
operation	是	String	是否进行桶授权。 取值如下： <ul style="list-style-type: none"> 0: 取消授权。 1: 授权。

响应参数

状态码：403

表 5-23 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

授权点播服务OBS桶权限，或者取消授权OBS桶权限

```
PUT https://{endpoint}/v1.0/{project_id}/asset/authority
```

```
Content-Type: application/json
{
  "bucket": "bucket",
  "operation": "1"
}
```

响应示例

状态码：403

处理失败返回。

```
{
  "error_code": "VOD.10053",
```

```
"error_msg": "The request parameter is illegal, illegal field: {xx}."
}
```

状态码

状态码	描述
200	处理成功返回。
403	处理失败返回。

错误码

请参见[错误码](#)。

5.5 创建媒资：OBS 转存方式

功能介绍

若您在使用点播服务前，已经在OBS桶中存储了音视频文件，可以使用该接口将OBS桶中的音视频文件转存至点播服务，再使用点播服务音视频管理功能。

使用该接口前，需要提前完成下述操作：

- 参考VOD用户指南“创建用户并授权使用VOD”章节，授权用户使用VOD。
- 参考OBS用户指南“创建IAM用户并授权使用OBS”章节，授权用户使用OBS。
- 调用该接口前，您需要调用VOD[桶授权](#)接口，将存储音视频文件的OBS桶授权给点播服务使用。

接口约束

由于不同区域的云服务不能互连，所以待转存的OBS桶和点播服务必须在同一区域，如“华北-北京四”OBS桶中的音视频只能转存到“华北-北京四”点播服务中。

URI

POST /v1.0/{project_id}/asset/reproduction

表 5-24 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

请求参数

表 5-25 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

表 5-26 请求 Body 参数

参数	是否必选	参数类型	描述
video_type	是	String	转存的音视频文件类型。 取值如下： <ul style="list-style-type: none"> • 视频文件：MP4、TS、MOV、MXF、MPG、FLV、WMV、AVI、M4V、F4V、MPEG、3GP、ASF、MKV、HLS、RMVB、WEBM • 音频文件：MP3、OGG、WAV、WMA、APE、FLAC、AAC、AC3、MMF、AMR、M4A、M4R、WV、MP2 若上传格式为音频文件，则不支持转码、添加水印和字幕。 说明 当video_type选择HLS时，storage_mode（存储模式）需选择存储在租户桶，且输出路径设置为和输入路径在同一个目录。
title	是	String	媒资标题，长度不超过128个字节，UTF-8编码。
description	否	String	视频描述，长度不超过1024个字节。

参数	是否必选	参数类型	描述
category_id	否	Integer	<p>媒资分类ID。</p> <p>您可以调用创建媒资分类接口或在点播控制台的分类设置中创建对应的媒资分类，并获取分类ID。</p> <p>说明 若不设置或者设置为-1，则上传的音视频归类到系统预置的“其它”分类中。</p>
tags	否	String	<p>视频标签。</p> <p>单个标签不超过24个字节，最多不超过16个标签。</p> <p>多个用逗号分隔，UTF8编码。</p>
auto_publish	否	Integer	<p>是否自动发布。</p> <p>取值如下：</p> <ul style="list-style-type: none"> ● 0：表示不自动发布。 ● 1：表示自动发布。 <p>默认值：1。</p>
template_group_name	否	String	<p>转码模板组名称。</p> <p>若不为空，则使用指定的转码模板对上传的音视频进行转码，您可以在视频点播控制台配置转码模板，具体请参见《点播服务用户指南》中的“转码设置”。</p> <p>说明 若同时设置了“<code>template_group_name</code>”和“<code>workflow_name</code>”字段，则“<code>template_group_name</code>”字段生效。</p>
auto_encrypt	否	Integer	<p>是否自动加密。</p> <p>取值如下：</p> <ul style="list-style-type: none"> ● 0：表示不加密。 ● 1：表示需要加密。 <p>默认值：0。</p> <p>若设置为需要加密，则必须配置转码模板，且转码的输出格式是HLS。</p>

参数	是否必选	参数类型	描述
auto_preheat	否	Integer	是否自动预热到CDN。 取值如下： <ul style="list-style-type: none"> • 0：表示不自动预热。 • 1：表示自动预热。 默认值：0。
thumbnail	否	Thumbnail object	截图参数
review	否	Review object	媒资审核参数。 说明 仅“亚太-新加坡”点播服务支持此功能。
workflow_name	否	String	工作流名称。若不为空，则使用指定的工作流对上传的音视频进行处理，您可以在视频点播控制台配置工作流，具体请参见《点播服务用户指南》中的“工作流设置”。
input	是	File_addr object	媒资存储参数信息。
storage_mode	否	Integer	存储模式。 取值如下： <ul style="list-style-type: none"> • 0：表示视频拷贝到点播桶。 • 1：表示视频存储在租户桶。 默认值：0
output_bucket	否	String	输出桶名，“ storage_mode ”为1时必选。
output_path	否	String	输出路径名，“ storage_mode ”为1时必选。

表 5-27 Thumbnail

参数	是否必选	参数类型	描述
type	是	String	截图类型。 取值如下： <ul style="list-style-type: none"> time: 每次进行截图的间隔时间。 dots: 按照指定的时间点截图。 quantity: 按照指定张数，根据视频时长等分视频截图。
quantity	否	Integer	type 取值为quantity时必填。 按照指定张数，根据视频时长等分视频截图。 取值范围：[1,10]之间的整数。
quantity_time	否	Integer	type 取值为quantity时选填。 按照指定时间间隔取指定张数截图。 取值范围：[0,2147483647]之间的整数。
time	否	Integer	根据时间间隔采样时的时间间隔值。单位：秒。 type 取值为time时。 默认值：12 取值范围：[0,100]之间的整数。
dots	否	Array of integers	type 取值为dots时必填。指定时间截图时的时间点数组。
cover_position	否	Integer	该值表示指定第几张截图作为封面。 默认值：1。
format	否	Integer	截图文件格式。 取值如下： <ul style="list-style-type: none"> 1: jpg。 默认值：1。

参数	是否必选	参数类型	描述
aspect_ratio	否	Integer	纵横比，图像缩放方式。 取值如下： <ul style="list-style-type: none"> 0: 自适应（保持原有宽高比）。 1: 16:9。 默认值：0。
max_length	否	Integer	截图最长边的尺寸。 单位：像素。 宽边尺寸按照该尺寸与原始视频像素等比缩放计算。 默认值：480。

表 5-28 Review

参数	是否必选	参数类型	描述
template_id	否	String	审核模板ID。您可以在视频点播控制台配置审核模板后获取，具体请参见《点播服务用户指南》中的“审核设置”。 说明 仅“亚太-新加坡”点播服务支持此功能。
interval	否	Integer	截图检测时间间隔，取值范围为(0,100]，该参数在请求参数中忽略。
politics	否	Integer	鉴政内容检测置信度，取值范围为-1或者[0,100]，该参数在请求参数中忽略。 置信度越高，说明审核结果越可信。未开启或设置为0时，表示未进行此项检测。politics、terrorism、porn三个参数不能同时为0。
terrorism	否	Integer	鉴恐内容的检测置信度，取值范围为-1或者[0,100]，该参数在请求参数中忽略。 置信度越高，说明审核结果越可信。未开启或设置为0时，表示未进行此项检测。politics、terrorism、porn三个参数不能同时为0。

参数	是否必选	参数类型	描述
porn	否	Integer	鉴黄内容的检测置信度，取值范围为-1或者[0,100]，该参数在请求参数中忽略。 置信度越高，说明审核结果越可信。未开启或设置为0时，表示未进行此项检测。politics、terrorism、porn三个参数不能同时为0。

表 5-29 File_addr

参数	是否必选	参数类型	描述
bucket	是	String	OBS的bucket名称。
location	是	String	桶所在的区域名，如“华北-北京四”的区域名为“cn-north-4”，创建的桶所在区域必须和点播服务所在区域保持一致。
object	是	String	文件的存储路径。

响应参数

状态码：200

表 5-30 响应 Body 参数

参数	参数类型	描述
asset_id	String	VOD分配的媒资ID，仅支持查询，不支持修改。

状态码：400

表 5-31 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

- 转存存储在OBS桶中的HLS视频至点播服务中（ts文件需和m3u8文件在同一目录下）

```
POST https://{endpoint}/v1.0/{project_id}/asset/reproduction
```

```
Content-Type: application/json
{
  "video_type": "HLS",
  "title": "tittle",
  "auto_publish": 1,
  "input": {
    "bucket": "bucket_name",
    "location": "cn-north-4",
    "object": "test/hls/test_video.m3u8"
  },
  "storage_mode": 1,
  "output_bucket": "bucket_name",
  "output_path": "test/hls"
}
```

- 转存存储在OBS桶中的音视频文件至点播服务中

```
POST https://{endpoint}/v1.0/{project_id}/asset/reproduction
```

```
Content-Type: application/json
{
  "input": {
    "bucket": "bucket",
    "location": "cn-north-4",
    "object": "path"
  },
  "title": "title",
  "description": "des",
  "category_id": -1,
  "tags": "test",
  "video_type": "MP4",
  "auto_publish": 1,
  "template_group_name": "tempName"
}
```

响应示例

状态码：200

处理成功返回。

```
{
  "asset_id" : "f488337c31c8e4622f1590735b134c65"
}
```

状态码：400

处理失败返回。

```
{
  "error_code" : "VOD.10003",
  "error_msg" : "The specified key does not exist."
}
```

状态码

状态码	描述
200	处理成功返回。

状态码	描述
400	处理失败返回。

错误码

请参见[错误码](#)。

5.6 创建媒资：URL 拉取注入（公测中）

功能介绍

通过音视频源文件的URL，离线拉取音视频文件至点播服务。如果音视频文件大小在5TB以下，可以使用这种方式。如果文件大小超出5TB，需要通过搬迁工具上传至点播服务，请提交工单获取工具和相应的操作指导。

接口约束

- 用户调用该API一次最多支持拉取16条音视频。
- URL拉取m3u8文件时，URL中不支持携带防盗链等鉴权参数。

URI

POST /v1.0/{project_id}/asset/upload_by_url

表 5-32 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目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	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。

参数	是否必选	参数类型	描述
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

表 5-34 请求 Body 参数

参数	是否必选	参数类型	描述
upload_metadata	是	Array of UploadMetadataByUrl objects	待拉取创建的媒资元数据

表 5-35 UploadMetadataByUrl

参数	是否必选	参数类型	描述
video_type	是	String	上传音视频文件的格式。 取值如下： <ul style="list-style-type: none"> • 视频文件：MP4、TS、MOV、MXF、MPG、FLV、WMV、AVI、M4V、F4V、MPEG、3GP、ASF、MKV、WEBM、M3U8 • 音频文件：MP3、OGG、WAV、WMA、APE、FLAC、AAC、AC3、MMF、AMR、M4A、M4R、WV、MP2 若上传格式为音频文件，则不支持转码、添加水印和字幕。
title	是	String	媒资标题，长度不超过128个字节，UTF-8编码。
url	是	String	待拉取音视频文件的URL。 说明 URL必须以扩展名结尾，暂只支持http和https协议，其中HTTP方式存在安全风险，建议使用HTTPS。不建议您将认证凭据等敏感信息写在URL。
description	否	String	视频描述，长度不超过1024个字节。

参数	是否必选	参数类型	描述
category_id	否	Integer	<p>媒资分类ID。</p> <p>您可以调用创建媒资分类接口或在点播控制台的分类设置中创建对应的媒资分类，并获取分类ID。</p> <p>说明 若不设置或者设置为-1，则上传的音视频归类到系统预置的“其它”分类中。</p>
tags	否	String	<p>视频标签。</p> <p>单个标签不超过24个字节，最多不超过16个标签。</p> <p>多个用逗号分隔，UTF8编码。</p>
auto_publish	否	Integer	<p>是否自动发布。</p> <p>取值如下：</p> <ul style="list-style-type: none"> ● 0：表示不自动发布。 ● 1：表示自动发布。 <p>默认值：1。</p>
template_group_name	否	String	<p>转码模板组名称。</p> <p>若不为空，则使用指定的转码模板对上传的音视频进行转码，您可以在视频点播控制台配置转码模板，具体请参见《点播服务用户指南》中的“转码设置”。</p> <p>说明 若同时设置了“template_group_name”和“workflow_name”字段，则“template_group_name”字段生效。</p>
auto_encrypt	否	Integer	<p>是否自动加密。</p> <p>取值如下：</p> <ul style="list-style-type: none"> ● 0：表示不加密。 ● 1：表示需要加密。 <p>默认值：0。若设置为需要加密，则必须配置转码模板，且转码的输出格式是HLS。</p>
auto_preheat	否	Integer	<p>是否自动预热到CDN。</p> <p>取值如下：</p> <ul style="list-style-type: none"> ● 0：表示不自动预热。 ● 1：表示自动预热。 <p>默认值：0。</p>

参数	是否必选	参数类型	描述
thumbnail	否	Thumbnail object	截图参数
review	否	Review object	媒资审核参数。 说明 仅“亚太-新加坡”点播服务支持此功能。
workflow_name	否	String	工作流名称。 若不为空，则使用指定的工作流对上传的音视频进行处理，您可以在视频点播控制台配置工作流，具体请参见《点播服务用户指南》中的“工作流设置”。

表 5-36 Thumbnail

参数	是否必选	参数类型	描述
type	是	String	截图类型。 取值如下： <ul style="list-style-type: none"> time：每次进行截图的间隔时间。 dots：按照指定的时间点截图。 quantity：按照指定张数，根据视频时长等分视频截图。
quantity	否	Integer	type 取值为quantity时必填。 按照指定张数，根据视频时长等分视频截图。 取值范围：[1,10]之间的整数。
quantity_time	否	Integer	type 取值为quantity时选填。 按照指定时间间隔取指定张数截图。 取值范围：[0,2147483647]之间的整数。
time	否	Integer	根据时间间隔采样时的时间间隔值。单位：秒。 type 取值为time时。 默认值：12 取值范围：[0,100]之间的整数。

参数	是否必选	参数类型	描述
dots	否	Array of integers	type 取值为dots时必填。指定时间截图时的时间点数组。
cover_position	否	Integer	该值表示指定第几张截图作为封面。 默认值：1。
format	否	Integer	截图文件格式。 取值如下： • 1: jpg。 默认值：1。
aspect_ratio	否	Integer	纵横比，图像缩放方式。 取值如下： • 0: 自适应（保持原有宽高比）。 • 1: 16:9。 默认值：0。
max_length	否	Integer	截图最长边的尺寸。 单位：像素。 宽边尺寸按照该尺寸与原始视频像素等比缩放计算。 默认值：480。

表 5-37 Review

参数	是否必选	参数类型	描述
template_id	否	String	审核模板ID。您可以在视频点播控制台配置审核模板后获取，具体请参见《点播服务用户指南》中的“审核设置”。 说明 仅“亚太-新加坡”点播服务支持此功能。
interval	否	Integer	截图检测时间间隔，取值范围为(0,100]，该参数在请求参数中忽略。

参数	是否必选	参数类型	描述
politics	否	Integer	鉴政内容检测置信度，取值范围为-1或者[0,100]，该参数在请求参数中忽略。 置信度越高，说明审核结果越可信。未开启或设置为0时，表示未进行此项检测。politics、terrorism、porn三个参数不能同时为0。
terrorism	否	Integer	鉴恐内容的检测置信度，取值范围为-1或者[0,100]，该参数在请求参数中忽略。 置信度越高，说明审核结果越可信。未开启或设置为0时，表示未进行此项检测。politics、terrorism、porn三个参数不能同时为0。
porn	否	Integer	鉴黄内容的检测置信度，取值范围为-1或者[0,100]，该参数在请求参数中忽略。 置信度越高，说明审核结果越可信。未开启或设置为0时，表示未进行此项检测。politics、terrorism、porn三个参数不能同时为0。

响应参数

状态码：200

表 5-38 响应 Body 参数

参数	参数类型	描述
upload_assets	Array of UploadAsset objects	待拉取创建的媒资元数据

表 5-39 UploadAsset

参数	参数类型	描述
url	String	待拉取音视频文件的URL。
asset_id	String	新创建媒资的媒资id

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

状态码：400

表 5-40 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

创建媒资，使用URL拉取方式。基于音视频源文件URL，将音视频文件离线拉取上传到点播服务

```
POST https://{endpoint}/v1.0/{project_id}/asset/upload_by_url
Content-Type: application/json
{
  "upload_metadatas": [
    {
      "url": "https://mpc-test.obs.cn-north-4.myhuaweicloud.com/Avatar_480P.mp4",
      "title": "Avatar test test",
      "video_type": "MP4",
      "description": "Avatar, test",
      "category_id": 1,
      "tags": "mytags",
      "auto_publish": 1
    },
    {
      "url": "https://mpc-test.obs.cn-north-4.myhuaweicloud.com/Avatar_720.mp4",
      "title": "Avatar test test",
      "video_type": "MP4",
      "description": "Avatar, test",
      "category_id": 1,
      "tags": "mytags",
      "auto_publish": 1
    }
  ]
}
```

响应示例

状态码：200

处理成功返回。

```
{
  "upload_assets": [ {
    "url": "https://mpc-test.obs.cn-north-4.myhuaweicloud.com/Avatar_480P.mp4",
    "asset_id": "f488337c31c8e4622f1590735b134c65",
    "error_code": null,

```

```
"error_msg": null
}, {
  "url": "https://mpc-test.obs.cn-north-4.myhuaweicloud.com/Avatar_720.mp4",
  "asset_id": "f488337c31c8e4622f1590525b134c65",
  "error_code": null,
  "error_msg": null
}]
}
```

状态码：400

处理失败返回。

```
{
  "error_code": "VOD.10003",
  "error_msg": "The specified key does not exist."
}
```

状态码

状态码	描述
200	处理成功返回。
400	处理失败返回。

错误码

请参见[错误码](#)。

5.7 上传检验

功能介绍

校验媒资文件是否已存储于视频点播服务中。

URI

GET /v1.0/{project_id}/asset/duplication

表 5-41 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

表 5-42 Query 参数

参数	是否必选	参数类型	描述
size	是	Integer	文件大小

参数	是否必选	参数类型	描述
md5	是	String	文件MD5。 建议参考《API参考》附录下生成MD5值的上传校验示例。

请求参数

表 5-43 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

响应参数

状态码：200

表 5-44 响应 Body 参数

参数	参数类型	描述
is_duplicated	Integer	是否重复。 取值如下： <ul style="list-style-type: none"> 0：表示不重复。 1：表示重复。
asset_ids	Array of strings	重复的媒资ID

状态码：400

表 5-45 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。

参数	参数类型	描述
error_msg	String	错误描述。

请求示例

校验媒资文件是否已存储于视频点播服务中

```
GET https://{endpoint}/v1.0/{project_id}/asset/duplication?md5={md5}&size={size}
```

响应示例

状态码：200

处理成功返回。

```
{  
  "is_duplicated" : 1,  
  "asset_ids" : [ "aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa" ]  
}
```

状态码：400

处理失败返回。

```
{  
  "error_code" : "VOD.10053",  
  "error_msg" : "The request parameter is illegal, illegal field: {xx}."  
}
```

状态码

状态码	描述
200	处理成功返回。
400	处理失败返回。

错误码

请参见[错误码](#)。

6 媒资处理

6.1 视频更新

功能介绍

媒资创建后，单独上传封面、更新视频文件或更新已有封面。如果是更新视频文件，更新完后要通过[确认媒资上传](#)接口通知点播服务。如果是更新封面或单独上传封面，则不需通知。更新视频可以使用分段上传，具体方式可以参考[示例2：媒资分段上传（20M以上）](#)。

接口约束

视频更新接口只支持更新下载失败、已取消和上传失败状态的媒资
请求Body参数的设置要求，如下所示：

- 参数video_name、cover_type、subtitles必须至少设置一个。
- 如果更新视频，参数video_name、video_type为必填项，不能为空值。
- 如果更新封面，参数cover_type为必填项，不能为空值，如果携带cover_id，则必须填0。
- 如果上传字幕文件，参数subtitles的id、type和language为必填项，不能为空值。

如果未按照上述要求设置参数，将无法返回接口响应消息。

URI

PUT /v1.0/{project_id}/asset

表 6-1 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

请求参数

表 6-2 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

表 6-3 请求 Body 参数

参数	是否必选	参数类型	描述
asset_id	是	String	VOD分配的媒资ID，仅支持查询，不支持修改。
video_md5	否	String	视频文件MD5值。建议参考《API参考》附录下生成MD5值的媒资上传和更新示例。
video_name	否	String	视频文件名。 文件名后缀为可选。
video_type	否	String	视频文件类型。 取值为MP4、TS、MOV、MXF、MPG、FLV、WMV、AVI、M4V、F4V、MPEG、3GP、ASF、MKV
cover_id	否	Integer	封面ID。 取值范围：[0,7]。 当前只支持一张封面，只能设置为0。
cover_type	否	String	封面图片格式类型。 取值如下： <ul style="list-style-type: none"> ● JPG ● PNG
cover_md5	否	String	封面文件的MD5值。

参数	是否必选	参数类型	描述
subtitles	否	Array of Subtitle objects	字幕文件信息

表 6-4 Subtitle

参数	是否必选	参数类型	描述
id	是	Integer	字幕id。 取值范围：[1,16]。
type	是	String	字幕文件类型，目前暂只支持“SRT”和“VTT”。
language	是	String	字幕语言类型。
name	否	String	字幕文件名称。
md5	否	String	字幕文件的MD5值。
description	否	String	字幕描述。

响应参数

状态码：200

表 6-5 响应 Body 参数

参数	参数类型	描述
asset_id	String	VOD分配的媒资ID，仅支持查询，不支持修改。
video_upload_url	String	<p>视频上传地址。</p> <p>用于客户端上传不超过20MB的小视频文件（<=20MB）。该URL中携带了临时授权信息，当文件大于20MB时，需要采用分段方式上传。</p> <p>说明 您可以使用PUT请求向“video_upload_url”中上传视频文件。其中，“Content-Type”需要根据上传的视频文件类型进行设置，如下所示：视频文件：video/视频格式，如：“video/mp4”。</p>

参数	参数类型	描述
cover_upload_url	String	封面上传地址，用于上传封面文件。 说明 您可以使用PUT请求向“cover_upload_url”中上传封面图片。其中，“Content-Type”需要根据上传的封面文件类型进行设置，如下所示：图片文件：image/图片格式，如：“image/png”。
subtitle_upload_urls	Array of strings	字幕上传地址，用于上传字幕。 说明 您可以使用PUT请求向“subtitle_upload_urls”中上传字幕文件。其中，“Content-Type”需要根据上传的字幕文件类型进行设置，如下所示：字幕文件：application/octet-stream。

状态码：403

表 6-6 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

更新媒资

PUT https://{endpoint}/v1.0/{project_id}/asset

```
Content-Type: application/json
{
  "asset_id": "f488337c31c8e4622f1590735b134c65",
  "cover_id": "0",
  "cover_type": "JPG",
  "subtitles": [
    {
      "id": 1,
      "language": "CN",
      "type": "SRT",
      "md5": "SqcyFjJZoDZaP8oKIY6rgQ==",
      "description": "AAAAA"
    }
  ]
}
```

响应示例

状态码：200

处理成功返回。


```
{
  "asset_id": "f488337c31c8e4622f1590735b134c65",
  "cover_upload_url": "https://obs.cn-north-4.myhuaweicloud.com:443/obs-vod-1/%7Bproject_id%7D/f488337c31c8e4622f1590735b134c65/cover/Cover0.jpg?AWSAccessKeyId=CBN2J*****0RCSN&Expires=1518148410&Signature=kZYh0hEos2V*****AHGyXA%3D",
  "subtitle_upload_urls": [ "https://obs-vod-1.obs.cn-north-4.myhuaweicloud.com:443/14ce1d4437164aba8b364ce15866154e/53a018d2dc53ca07eb5a07a839205c9d/subtitle/1.srt?AWSAccessKeyId=CBN2J*****0RCSN&Expires=1534760131&Signature=kZYh0hEos2V*****AHGyXA%3D" ]
}
```

状态码：403

处理失败返回。

```
{
  "error_code": "VOD.10062",
  "error_msg": "Media asset or resource does not exist, please check."
}
```

状态码

状态码	描述
200	处理成功返回。
403	处理失败返回。

错误码

请参见[错误码](#)。

6.2 媒资处理

功能介绍

实现视频转码、截图、加密等处理。既可以同时启动多种操作，也可以只启动一种操作。

URI

POST /v1.0/{project_id}/asset/process

表 6-7 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

请求参数

表 6-8 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

表 6-9 请求 Body 参数

参数	是否必选	参数类型	描述
asset_id	是	String	VOD分配的媒资ID，仅支持查询，不支持修改。
template_group_name	否	String	转码模板组名称。 说明 若不为空，则使用指定的转码模板对上传的音视频进行转码，您可以在视频点播控制台配置转码模板，具体请参见《点播服务用户指南》中的“转码设置”。
auto_encrypt	否	Integer	是否自动加密。 取值如下： <ul style="list-style-type: none"> 0：表示不加密。 1：表示需要加密。 默认值：0。 加密与转码必须要一起进行，当需要加密时，转码参数不能为空，且转码输出格式必须要为HLS。
thumbnail	否	Thumbnail object	截图参数

参数	是否必选	参数类型	描述
subtitle_id	否	Array of integers	字幕文件ID。 说明 <ul style="list-style-type: none"> 仅调用接口创建媒资：上传方式和视频更新，且设置请求参数“subtitles”时，该参数设置才生效。 如果参数取值为空，默认选择最后一次上传的字幕压制至视频流中。 视频流压制的字幕文件仅支持SRT格式。

表 6-10 Thumbnail

参数	是否必选	参数类型	描述
type	是	String	截图类型。 取值如下： <ul style="list-style-type: none"> time：每次进行截图的间隔时间。 dots：按照指定的时间点截图。 quantity：按照指定张数，根据视频时长等分视频截图。
quantity	否	Integer	type 取值为quantity时必填。按照指定张数，根据视频时长等分视频截图。 取值范围：[1,10]之间的整数。
quantity_time	否	Integer	type 取值为quantity时选填。按照指定时间间隔取指定张数截图。 取值范围：[0,2147483647]之间的整数。
time	否	Integer	根据时间间隔采样时的时间间隔值。单位：秒。 type 取值为time时。 默认值：12 取值范围：[0,100]之间的整数。
dots	否	Array of integers	type 取值为dots时必填。指定时间截图时的时间点数组。

参数	是否必选	参数类型	描述
cover_position	否	Integer	该值表示指定第几张截图作为封面。 默认值：1。
format	否	Integer	截图文件格式。 取值如下： • 1: jpg。 默认值：1。
aspect_ratio	否	Integer	纵横比，图像缩放方式。 取值如下： • 0: 自适应（保持原有宽高比）。 • 1: 16:9。 默认值：0。
max_length	否	Integer	截图最长边的尺寸。 单位：像素。 宽边尺寸按照该尺寸与原始视频像素等比缩放计算。 默认值：480。

响应参数

状态码：202

表 6-11 响应 Body 参数

参数	参数类型	描述
asset_id	String	VOD分配的媒资ID，仅支持查询，不支持修改。

状态码：403

表 6-12 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

媒资转码处理

POST https://{endpoint}/v1.0/{project_id}/asset/process

Content-Type: application/json

```
{
  "asset_id": "b4f39691d66cc0ef75d62ee567146e11",
  "template_group_name": "test",
  "auto_encrypt": 0,
  "thumbnail": {
    "type": "time",
    "time": 12
  },
  "subtitle_id": [
    1
  ]
}
```

响应示例

状态码：202

处理成功返回。

```
{
  "asset_id": "b4f39691d66cc0ef75d62ee567146e11"
}
```

状态码：403

处理失败返回。

```
{
  "error_code": "VOD.10062",
  "error_msg": "Media asset or resource does not exist, please check."
}
```

状态码

状态码	描述
202	处理成功返回。
403	处理失败返回。

错误码

请参见[错误码](#)。

6.3 取消媒资转码任务

功能介绍

取消媒资转码任务，只能取消排队中的转码任务。

URI

DELETE /v1.0/{project_id}/asset/process

表 6-13 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

表 6-14 Query 参数

参数	是否必选	参数类型	描述
asset_id	是	String	VOD分配的媒资ID，仅支持查询，不支持修改。

请求参数

表 6-15 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

响应参数

状态码：400

表 6-16 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

取消排队中的媒资转码任务

```
DELETE https://{endpoint}/v1.0/{project_id}/asset/process?asset_id={asset_id}
```

响应示例

状态码：400

处理失败返回。

```
{
  "error_code" : "VOD.10083",
  "error_msg" : "The current media asset status does not support this operation."
}
```

状态码

状态码	描述
204	处理成功返回。
400	处理失败返回。

错误码

请参见[错误码](#)。

6.4 音频提取

功能介绍

本接口为异步接口，创建音频提取任务下发成功后会返回asset_id和提取的audio_asset_id，但此时音频提取任务并没有立即完成，可通过消息订阅界面配置的音频提取完成事件来获取音频提取任务完成与否。

URI

POST /v1.0/{project_id}/asset/extract_audio

表 6-17 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

请求参数

表 6-18 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

表 6-19 请求 Body 参数

参数	是否必选	参数类型	描述
asset_id	是	String	VOD分配的媒资ID，仅支持查询，不支持修改。
parameter	否	Parameter object	需要提取音频的参数信息。

表 6-20 Parameter

参数	是否必选	参数类型	描述
format	否	String	封装格式。 取值如下： <ul style="list-style-type: none"> • MP3 • AAC

响应参数

状态码：202

表 6-21 响应 Body 参数

参数	参数类型	描述
asset_id	String	视频源媒资ID。

参数	参数类型	描述
audio_asset_id	String	提取的音频媒资ID。

状态码：400

表 6-22 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

提取音频

POST https://{endpoint}/v1.0/{project_id}/asset/extract_audio

```
Content-Type: application/json
{
  "asset_id": "3e1cd21131a94525be55acf65888bf46",
  "parameter": {
    "format": "MP3"
  }
}
```

响应示例

状态码：202

处理成功返回

```
{
  "asset_id": "f488337c31c8e4622f1590735b134c65",
  "audio_asset_id": "5412"
}
```

状态码：400

处理失败返回

```
{
  "error_code": "VOD.10062",
  "error_msg": "Media asset or resource does not exist, please check."
}
```

状态码

状态码	描述
202	处理成功返回
400	处理失败返回

错误码

请参见[错误码](#)。

6.5 取消提取音频任务

功能介绍

取消提取音频任务，只有排队中的提取音频任务才可以取消。

URI

DELETE /v1.0/{project_id}/asset/extract_audio

表 6-23 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

表 6-24 Query 参数

参数	是否必选	参数类型	描述
asset_id	是	String	VOD分配的媒资ID，仅支持查询，不支持修改。

请求参数

表 6-25 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

响应参数

状态码：400

表 6-26 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

取消音频提取任务

```
DELETE https://{endpoint}/v1.0/{project_id}/asset/extract_audio?asset_id={asset_id}
```

响应示例

状态码：400

处理失败返回。

```
{  
  "error_code": "VOD.10083",  
  "error_msg": "The current media asset status does not support this operation."  
}
```

状态码

状态码	描述
204	处理成功返回。
400	处理失败返回。

错误码

请参见[错误码](#)。

6.6 创建审核媒资任务

功能介绍

对上传的媒资进行审核。审核后，可以调用[查询媒资详细信息](#)接口查看审核结果。

接口约束

目前仅“华北-北京一”、“华北-北京四”区域的点播服务支持此功能。

URI

POST /v1.0/{project_id}/asset/review

表 6-27 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

请求参数

表 6-28 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

表 6-29 请求 Body 参数

参数	是否必选	参数类型	描述
asset_id	是	String	VOD分配的媒资ID，仅支持查询，不支持修改。
review	是	Review object	媒资审核参数。 说明 仅“亚太-新加坡”点播服务支持此功能。

表 6-30 Review

参数	是否必选	参数类型	描述
template_id	否	String	审核模板ID。您可以在视频点播控制台配置审核模板后获取，具体请参见《点播服务用户指南》中的“审核设置”。 说明 仅“亚太-新加坡”点播服务支持此功能。
interval	否	Integer	截图检测时间间隔，取值范围为(0,100]，该参数在请求参数中忽略。
politics	否	Integer	鉴政内容检测置信度，取值范围为-1或者[0,100]，该参数在请求参数中忽略。 置信度越高，说明审核结果越可信。未开启或设置为0时，表示未进行此项检测。politics、terrorism、porn三个参数不能同时为0。
terrorism	否	Integer	鉴恐内容的检测置信度，取值范围为-1或者[0,100]，该参数在请求参数中忽略。 置信度越高，说明审核结果越可信。未开启或设置为0时，表示未进行此项检测。politics、terrorism、porn三个参数不能同时为0。
porn	否	Integer	鉴黄内容的检测置信度，取值范围为-1或者[0,100]，该参数在请求参数中忽略。 置信度越高，说明审核结果越可信。未开启或设置为0时，表示未进行此项检测。politics、terrorism、porn三个参数不能同时为0。

响应参数

状态码：200

表 6-31 响应 Body 参数

参数	参数类型	描述
asset_id	String	VOD分配的媒资ID，仅支持查询，不支持修改。
review	Review object	媒资审核参数。 说明 仅“亚太-新加坡”点播服务支持此功能。

表 6-32 Review

参数	参数类型	描述
template_id	String	审核模板ID。您可以在视频点播控制台配置审核模板后获取，具体请参见《点播服务用户指南》中的“审核设置”。 说明 仅“亚太-新加坡”点播服务支持此功能。
interval	Integer	截图检测时间间隔，取值范围为(0,100]，该参数在请求参数中忽略。
politics	Integer	鉴政内容检测置信度，取值范围为-1或者[0,100]，该参数在请求参数中忽略。置信度越高，说明审核结果越可信。未开启或设置为0时，表示未进行此项检测。politics、terrorism、porn三个参数不能同时为0。
terrorism	Integer	鉴恐内容的检测置信度，取值范围为-1或者[0,100]，该参数在请求参数中忽略。置信度越高，说明审核结果越可信。未开启或设置为0时，表示未进行此项检测。politics、terrorism、porn三个参数不能同时为0。
porn	Integer	鉴黄内容的检测置信度，取值范围为-1或者[0,100]，该参数在请求参数中忽略。置信度越高，说明审核结果越可信。未开启或设置为0时，表示未进行此项检测。politics、terrorism、porn三个参数不能同时为0。

状态码：400

表 6-33 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

审核上传的媒资

POST https://{endpoint}/v1.0/{project_id}/asset/review

Content-Type: application/json

```
{
  "asset_id": "3e1cd21131a94525be55acf65888bf46",
  "review": {
    "template_id": "c80e56dad8542e8a1b7c2224dd6733a"
  }
}
```

响应示例

状态码：200

处理成功返回。

```
{
  "asset_id": "ede1391f8be15b4bcf99099c8d437c00",
  "review": {
    "template_id": "c80e56dad8542e8a1b7c2224dd6733a",
    "interval": 5,
    "politics": 0,
    "terrorism": 0,
    "porn": 1
  }
}
```

状态码：400

处理失败返回。

```
{
  "error_code": "VOD.10062",
  "error_msg": "Media asset or resource does not exist, please check."
}
```

状态码

状态码	描述
200	处理成功返回。
400	处理失败返回。

错误码

请参见[错误码](#)。

6.7 设置封面

功能介绍

将视频截图生成的某张图片设置成封面。

URI

PUT /v1.0/{project_id}/asset/cover

表 6-34 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

请求参数

表 6-35 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

表 6-36 请求 Body 参数

参数	是否必选	参数类型	描述
thumbnailUrl	是	String	截图文件的URL。 需要根据媒资ID调用 查询媒资详细信息 接口获取媒资的截图文件URL。

响应参数

状态码：400

表 6-37 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

设置视频封面，截取视频截图作为封面

```
PUT https://{endpoint}/v1.0/{project_id}/asset/cover
```

```
Content-Type: application/json
```

```
{
  "thumbnailUrl": "
  https://355.cdn-vod.huaweicloud.com/shield/asset/6e76f92034ceae2fcdcef9413221511f/snapshot/sample/
  0.jpg"
}
```

响应示例

状态码：400

处理失败返回。

```
{
  "error_code" : "VOD.10062",
  "error_msg" : "Media asset or resource does not exist, please check."
}
```

状态码

状态码	描述
204	处理成功返回。
400	处理失败返回。

错误码

请参见[错误码](#)。

7 媒资管理

7.1 删除媒资

功能介绍

删除媒资。

接口约束

正在处理中的媒资不支持删除，包括转码中、审核中、截图中等。

URI

DELETE /v1.0/{project_id}/asset

表 7-1 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

表 7-2 Query 参数

参数	是否必选	参数类型	描述
asset_id	是	Array of strings	媒资ID，支持一次删除多个媒资，批量删除时以逗号分隔。取值范围[1,10]。
delete_type	否	String	删除类型，当值为origin时只删除源文件，保留转码后文件，不填的话走默认删除整个媒资。

请求参数

表 7-3 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

响应参数

状态码：200

表 7-4 响应 Body 参数

参数	参数类型	描述
delete_result_array	Array of DeleteResult objects	删除媒资任务的处理结果。

表 7-5 DeleteResult

参数	参数类型	描述
asset_id	String	VOD分配的媒资ID，仅支持查询，不支持修改。
status	String	删除状态。 取值如下： <ul style="list-style-type: none"> DELETED：已删除。 FAILED：删除失败。

状态码：403

表 7-6 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

删除媒资

```
DELETE https://{endpoint}/v1.0/{project_id}/asset?asset_id={asset_id}
```

响应示例

状态码：200

处理成功返回。

```
{
  "delete_result_array": [ {
    "asset_id": "f488337c31c8e4622f1590735b134c65",
    "status": "DELETED"
  } ]
}
```

状态码：403

处理失败返回。

```
{
  "error_code": "VOD.10062",
  "error_msg": "Media asset or resource does not exist, please check."
}
```

状态码

状态码	描述
200	处理成功返回。
403	处理失败返回。

错误码

请参见[错误码](#)。

7.2 媒资发布

功能介绍

将媒资设置为发布状态。支持批量发布。

URI

POST /v1.0/{project_id}/asset/status/publish

表 7-7 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

请求参数

表 7-8 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

表 7-9 请求 Body 参数

参数	是否必选	参数类型	描述
asset_id	是	Array of strings	VOD分配的媒资ID，仅支持查询，不支持修改。

响应参数

状态码：200

表 7-10 响应 Body 参数

参数	参数类型	描述
asset_info_array	Array of AssetInfo objects	发布的媒资信息。

表 7-11 AssetInfo

参数	参数类型	描述
asset_id	String	VOD分配的媒资ID，仅支持查询，不支持修改。
is_multi_transcode	Boolean	是否是多转码模式
status	String	<p>媒资状态。</p> <p>取值如下：</p> <ul style="list-style-type: none"> • UNCREATED：未创建（媒资ID不存在） • DELETED：已删除 • CANCELLED：上传取消 • SERVER_ERROR：上传失败（点播服务端故障） • UPLOAD_FAILED：上传失败（向OBS上传失败） • CREATING：创建中 • PUBLISHED：已发布 • WAITING_TRANSCODE：待发布（转码排队中） • TRANSCODING：待发布（转码中） • TRANSCODE_FAILED：待发布（转码失败） • TRANSCODE_SUCCEED：待发布（转码成功） • CREATED：待发布（未转码） • NO_ASSET：该媒资不存在 • DELETING：正在删除 • DELETE_FAILED：删除失败 • OBS_CREATING：OBS转存方式创建中 • OBS_CREATE_FAILED：OBS转存失败 • OBS_CREATE_SUCCESS：OBS转存成功
description	String	<p>媒资子状态或描述信息。</p> <ul style="list-style-type: none"> • 对于媒资异常场景，描述具体的异常原因。 • 对于正常场景，描述媒资的处理信息。
base_info	BaseInfo object	媒资基本信息。

参数	参数类型	描述
play_info_array	Array of PlayInfo objects	转码文件的播放信息。 <ul style="list-style-type: none"> • HLS或DASH: 此数组的成员个数为n+1, n为转码输出路数。 • MP4: 此数组的成员个数为n, n为转码输出路数。

表 7-12 BaseInfo

参数	参数类型	描述
title	String	媒资标题。 长度不超过128个字节, UTF8编码。
video_name	String	媒资文件名。
description	String	媒资描述。 长度不超过1024个字节。
category_id	Long	媒资分类id。
category_name	String	媒资分类名称。
create_time	String	媒资创建时间。 格式为yyyymmddhhmmss。必须是与时区无关的UTC时间。
last_modified	String	媒资最近修改时间。 格式为yyyymmddhhmmss。必须是与时区无关的UTC时间。
video_type	String	音视频文件类型。 取值如下: <ul style="list-style-type: none"> • 视频文件: MP4、TS、MOV、MXF、MPG、FLV、WMV、AVI、M4V、F4V、MPEG、3GP、ASF、MKV、WEBM • 音频文件: MP3、OGG、WAV、WMA、APE、FLAC、AAC、AC3、MMF、AMR、M4A、M4R、WV、MP2
tags	String	媒资标签。 单个标签不超过24个字节, 最多不超过16个标签。 多个用逗号分隔, UTF8编码。

参数	参数类型	描述
meta_data	MetaData object	视频的元数据信息。 经过视频解析后产生，包括封装格式、大小、分辨率、码率、帧率。
video_url	String	原始视频文件的访问地址。
sign_url	String	原视频文件的OBS临时访问地址。仅调用 查询指定媒资的详细信息 接口时才返回具体值。
cover_info_array	Array of CoverInfo objects	封面信息。
subtitle_info	Array of SubtitleInfo objects	字幕信息数组
source_path	File_addr object	媒资存储参数信息。
output_path	File_addr object	媒资存储参数信息。

表 7-13 CoverInfo

参数	参数类型	描述
cover_url	String	封面文件的下载地址。

表 7-14 SubtitleInfo

参数	参数类型	描述
url	String	字幕文件的下载地址
id	Integer	字幕文件id
type	String	字幕文件类型
language	String	字幕文件语言种类

表 7-15 File_addr

参数	参数类型	描述
bucket	String	OBS的bucket名称。

参数	参数类型	描述
location	String	桶所在的区域名，如“华北-北京四”的区域名为“cn-north-4”，创建的桶所在区域必须和点播服务所在区域保持一致。
object	String	文件的存储路径。

表 7-16 PlayInfo

参数	参数类型	描述
play_type	String	播放协议类型。 取值如下： <ul style="list-style-type: none"> • hls • dash • mp4
group_id	String	所属转码组Id
group_name	String	所属转码组名称
url	String	播放URL。
encrypted	Integer	标记流是否已被加密。 取值如下： <ul style="list-style-type: none"> • 0：表示未加密。 • 1：表示已被加密。 默认值：0。
meta_data	MetaData object	视频的元数据信息。 经过视频解析后产生，包括封装格式、大小、分辨率、码率、帧率。

表 7-17 MetaData

参数	参数类型	描述
pack_type	String	<p>视频封装格式。</p> <p>取值如下：</p> <ul style="list-style-type: none"> • MP4 • TS • MOV • MXF • MPG • FLV • WMV • MP3 • WMA • APE • FLAC • AAC • AC3 • MMF • AMR • M4A • M4R • OGG • WAV • WV • MP2 • AVI • F4V • M4V • MPEG • HLS • DASH

参数	参数类型	描述
codec	String	<p>视频编码格式。</p> <p>取值如下：</p> <ul style="list-style-type: none"> • MPEG-2 • MPEG-4 • H.264 • H.265 • WMV • Vorbis • AAC • AC-3 • AMR • APE • FLAC • MP3 • MP2 • WMA • PCM • ADPCM • WavPack <p>说明 如果codec返回unknown，说明不支持解析用户当前发送的音视频编码格式。</p>
duration	Long	<p>视频时长，单位：秒。</p> <p>若视频的原时长为非整数，则该字段值为原时长的向下取整。</p> <p>若视频的原时长小于1，则该字段值为1。</p>
duration_ms	Long	视频时长，单位毫秒。
video_size	Long	<p>视频文件大小。</p> <p>单位：字节。</p>
width	Long	<p>视频宽度（单位：像素）。</p> <ul style="list-style-type: none"> • 编码为H.264的取值范围：[32,3840]之间2的倍数。 • 编码为H.265的取值范围：[320,3840]之间4的倍数。

参数	参数类型	描述
hight	Long	视频高度（单位：像素）。 <ul style="list-style-type: none"> 编码为H.264的取值范围：[32,2160]之间2的倍数。 编码为H.265的取值范围：[240,2160]之间4的倍数。
height	Long	视频高度（单位：像素）
bit_rate	Long	视频平均码率。
frame_rate	Long	帧率（单位：帧每秒）。 取值如下： <ul style="list-style-type: none"> FRAMERATE_AUTO = 1, FRAMERATE_10 = 2, FRAMERATE_15 = 3, FRAMERATE_2397 = 4, // 23.97 fps FRAMERATE_24 = 5, FRAMERATE_25 = 6, FRAMERATE_2997 = 7, // 29.97 fps FRAMERATE_30 = 8, FRAMERATE_50 = 9, FRAMERATE_60 = 10 默认值：1。 单位：帧每秒。
quality	String	清晰度。 取值如下： <ul style="list-style-type: none"> FULL_HD：超高清 HD：高清 SD：标清 FLUENT：流畅 AD：自适应 2K 4K
audio_channels	Integer	音频的声道数。

状态码：403

表 7-18 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

发布媒资

POST https://{endpoint}/v1.0/{project_id}/asset/status/publish

Content-Type: application/json

```
{
  "asset_id": [
    "f488337c31c8e4622f1590735b134c65",
    "07ba4d46463355e800c2f42f628f0631"
  ]
}
```

响应示例

状态码：200

处理成功返回。

```
{
  "asset_info_array": [ {
    "asset_id": "f488337c31c8e4622f1590735b134c65",
    "status": "UNCREATED",
    "description": "Asset meta download fail, errorCode is VOD.100021010,\nerrorMsg is Internal server communication is abnormal",
    "base_info": {
      "title": "Avatar test test",
      "video_name": "Avatar_480P.mp4",
      "description": "Avatar, test",
      "category_id": -1,
      "category_name": "其他",
      "create_time": "20180209024019",
      "last_modified": "20180209024019",
      "video_type": "MP4",
      "tags": "mytags"
    }
  }, {
    "asset_id": "07ba4d46463355e800c2f42f628f0631",
    "status": " UNCREATED",
    "description": "no asset"
  }
]
```

状态码：403

处理失败返回。

```
{
  "error_code": "VOD.10062",
  "error_msg": "Media asset or resource does not exist, please check."
}
```

状态码

状态码	描述
200	处理成功返回。
403	处理失败返回。

错误码

请参见[错误码](#)。

7.3 媒资发布取消

功能介绍

将媒资设置为未发布状态。

URI

POST /v1.0/{project_id}/asset/status/unpublish

表 7-19 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

请求参数

表 7-20 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

表 7-21 请求 Body 参数

参数	是否必选	参数类型	描述
asset_id	是	Array of strings	VOD分配的媒资ID，仅支持查询，不支持修改。

响应参数

状态码：200

表 7-22 响应 Body 参数

参数	参数类型	描述
asset_info_array	Array of AssetInfo objects	媒资状态集合。

表 7-23 AssetInfo

参数	参数类型	描述
asset_id	String	VOD分配的媒资ID，仅支持查询，不支持修改。
is_multi_transcode	Boolean	是否是多转码模式

参数	参数类型	描述
status	String	<p>媒资状态。</p> <p>取值如下：</p> <ul style="list-style-type: none"> • UNCREATED: 未创建（媒资ID不存在） • DELETED: 已删除 • CANCELLED: 上传取消 • SERVER_ERROR: 上传失败（点播服务端故障） • UPLOAD_FAILED: 上传失败（向OBS上传失败） • CREATING: 创建中 • PUBLISHED: 已发布 • WAITING_TRANSCODE: 待发布（转码排队中） • TRANSCODING: 待发布（转码中） • TRANSCODE_FAILED: 待发布（转码失败） • TRANSCODE_SUCCEED: 待发布（转码成功） • CREATED: 待发布（未转码） • NO_ASSET: 该媒资不存在 • DELETING: 正在删除 • DELETE_FAILED: 删除失败 • OBS_CREATING: OBS转存方式创建中 • OBS_CREATE_FAILED: OBS转存失败 • OBS_CREATE_SUCCESS: OBS转存成功
description	String	<p>媒资子状态或描述信息。</p> <ul style="list-style-type: none"> • 对于媒资异常场景，描述具体的异常原因。 • 对于正常场景，描述媒资的处理信息。
base_info	BaseInfo object	媒资基本信息。
play_info_array	Array of PlayInfo objects	<p>转码文件的播放信息。</p> <ul style="list-style-type: none"> • HLS或DASH: 此数组的成员个数为n+1，n为转码输出路数。 • MP4: 此数组的成员个数为n，n为转码输出路数。

表 7-24 BaseInfo

参数	参数类型	描述
title	String	媒资标题。 长度不超过128个字节，UTF8编码。
video_name	String	媒资文件名。
description	String	媒资描述。 长度不超过1024个字节。
category_id	Long	媒资分类id。
category_name	String	媒资分类名称。
create_time	String	媒资创建时间。 格式为yyyymmddhhmmss。必须是与时区无关的UTC时间。
last_modified	String	媒资最近修改时间。 格式为yyyymmddhhmmss。必须是与时区无关的UTC时间。
video_type	String	音视频文件类型。 取值如下： <ul style="list-style-type: none"> • 视频文件：MP4、TS、MOV、MXF、MPG、FLV、WMV、AVI、M4V、F4V、MPEG、3GP、ASF、MKV、WEBM • 音频文件：MP3、OGG、WAV、WMA、APE、FLAC、AAC、AC3、MMF、AMR、M4A、M4R、WV、MP2
tags	String	媒资标签。 单个标签不超过24个字节，最多不超过16个标签。 多个用逗号分隔，UTF8编码。
meta_data	MetaData object	视频的元数据信息。 经过视频解析后产生，包括封装格式、大小、分辨率、码率、帧率。
video_url	String	原始视频文件的访问地址。
sign_url	String	原视频文件的OBS临时访问地址。仅调用 查询指定媒资的详细信息 接口时才返回具体值。

参数	参数类型	描述
cover_info_array	Array of CoverInfo objects	封面信息。
subtitle_info	Array of SubtitleInfo objects	字幕信息数组
source_path	File_addr object	媒资存储参数信息。
output_path	File_addr object	媒资存储参数信息。

表 7-25 CoverInfo

参数	参数类型	描述
cover_url	String	封面文件的下载地址。

表 7-26 SubtitleInfo

参数	参数类型	描述
url	String	字幕文件的下载地址
id	Integer	字幕文件id
type	String	字幕文件类型
language	String	字幕文件语言种类

表 7-27 File_addr

参数	参数类型	描述
bucket	String	OBS的bucket名称。
location	String	桶所在的区域名，如“华北-北京四”的区域名为“cn-north-4”，创建的桶所在区域必须和点播服务所在区域保持一致。
object	String	文件的存储路径。

表 7-28 PlayInfo

参数	参数类型	描述
play_type	String	播放协议类型。 取值如下： <ul style="list-style-type: none"> • hls • dash • mp4
group_id	String	所属转码组Id
group_name	String	所属转码组名称
url	String	播放URL。
encrypted	Integer	标记流是否已被加密。 取值如下： <ul style="list-style-type: none"> • 0：表示未加密。 • 1：表示已被加密。 默认值：0。
meta_data	MetaData object	视频的元数据信息。 经过视频解析后产生，包括封装格式、大小、分辨率、码率、帧率。

表 7-29 MetaData

参数	参数类型	描述
pack_type	String	<p>视频封装格式。</p> <p>取值如下：</p> <ul style="list-style-type: none"> • MP4 • TS • MOV • MXF • MPG • FLV • WMV • MP3 • WMA • APE • FLAC • AAC • AC3 • MMF • AMR • M4A • M4R • OGG • WAV • WV • MP2 • AVI • F4V • M4V • MPEG • HLS • DASH

参数	参数类型	描述
codec	String	<p>视频编码格式。</p> <p>取值如下：</p> <ul style="list-style-type: none"> • MPEG-2 • MPEG-4 • H.264 • H.265 • WMV • Vorbis • AAC • AC-3 • AMR • APE • FLAC • MP3 • MP2 • WMA • PCM • ADPCM • WavPack <p>说明 如果codec返回unknown，说明不支持解析用户当前发送的音视频编码格式。</p>
duration	Long	<p>视频时长，单位：秒。</p> <p>若视频的原时长为非整数，则该字段值为原时长的向下取整。</p> <p>若视频的原时长小于1，则该字段值为1。</p>
duration_ms	Long	视频时长，单位毫秒。
video_size	Long	<p>视频文件大小。</p> <p>单位：字节。</p>
width	Long	<p>视频宽度（单位：像素）。</p> <ul style="list-style-type: none"> • 编码为H.264的取值范围：[32,3840]之间2的倍数。 • 编码为H.265的取值范围：[320,3840]之间4的倍数。

参数	参数类型	描述
hight	Long	视频高度（单位：像素）。 <ul style="list-style-type: none"> 编码为H.264的取值范围：[32,2160]之间2的倍数。 编码为H.265的取值范围：[240,2160]之间4的倍数。
height	Long	视频高度（单位：像素）
bit_rate	Long	视频平均码率。
frame_rate	Long	帧率（单位：帧每秒）。 取值如下： <ul style="list-style-type: none"> FRAMERATE_AUTO = 1, FRAMERATE_10 = 2, FRAMERATE_15 = 3, FRAMERATE_2397 = 4, // 23.97 fps FRAMERATE_24 = 5, FRAMERATE_25 = 6, FRAMERATE_2997 = 7, // 29.97 fps FRAMERATE_30 = 8, FRAMERATE_50 = 9, FRAMERATE_60 = 10 默认值：1。 单位：帧每秒。
quality	String	清晰度。 取值如下： <ul style="list-style-type: none"> FULL_HD：超高清 HD：高清 SD：标清 FLUENT：流畅 AD：自适应 2K 4K
audio_channels	Integer	音频的声道数。

状态码：403

表 7-30 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

取消发布媒资

POST https://{endpoint}/v1.0/{project_id}/asset/status/unpublish

Content-Type: application/json

```
{
  "asset_id": [
    "f488337c31c8e4622f1590735b134c65",
    "07ba4d46463355e800c2f42f628f0631"
  ]
}
```

响应示例

状态码：200

处理成功返回。

```
{
  "asset_info_array": [ {
    "asset_id": "f488337c31c8e4622f1590735b134c65",
    "status": "UNCREATED",
    "description": "Asset meta download fail, errorCode is VOD.100021010,\nerrorMsg is Internal server communication is abnormal",
    "base_info": {
      "title": "Avatar test test",
      "video_name": "Avatar_480P.mp4",
      "description": "Avatar, test",
      "category_id": -1,
      "category_name": "其他",
      "create_time": "20180209024019",
      "last_modified": "20180209024019",
      "video_type": "MP4",
      "tags": "mytags"
    }
  }, {
    "asset_id": "07ba4d46463355e800c2f42f628f0631",
    "status": "NOASSET",
    "description": "no asset"
  }
]
```

状态码：403

处理失败返回。

```
{
  "error_code": "VOD.10062",
  "error_msg": "Media asset or resource does not exist, please check."
}
```

状态码

状态码	描述
200	处理成功返回。
403	处理失败返回。

错误码

请参见[错误码](#)。

7.4 查询媒资信息

功能介绍

查询媒资信息，支持指定媒资ID、分类、状态、起止时间查询。

URI

GET /v1.0/{project_id}/asset/info

表 7-31 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

表 7-32 Query 参数

参数	是否必选	参数类型	描述
asset_id	否	Array of strings	媒资id，最多同时查询10个媒资。

参数	是否必选	参数类型	描述
status	否	Array of strings	媒资状态。 取值如下： <ul style="list-style-type: none"> • UNCREATED: 未创建（媒资ID不存在） • DELETED: 已删除 • CANCELLED: 上传取消 • SERVER_ERROR: 上传失败（点播服务端故障） • UPLOAD_FAILED: 上传失败（向OBS上传失败） • CREATING: 创建中 • PUBLISHED: 已发布 • TRANSCODING: 待发布（转码中） • TRANSCODE_FAILED: 待发布（转码失败） • TRANSCODE_SUCCEED: 待发布（转码成功） • CREATED: 待发布（未转码）
transcodeStatus	否	Array of strings	转码状态 取值如下： <ul style="list-style-type: none"> • TRANSCODING: 转码中 • TRANSCODE_FAILED: 转码失败 • TRANSCODE_SUCCEED: 转码成功 • UN_TRANSCODE: 未转码 • WAITING_TRANSCODE: 等待转码
assetStatus	否	Array of strings	媒资状态。 取值如下： <ul style="list-style-type: none"> • PUBLISHED: 已发布 • CREATED: 未发布
start_time	否	String	起始时间，查询指定“asset_id”时，该参数无效。 格式为yyyymmddhhmmss。必须是与时区无关的UTC时间。

参数	是否必选	参数类型	描述
end_time	否	String	结束时间，查询指定“asset_id”时，该参数无效。 格式为yyyymmddhhmmss。必须是与时区无关的UTC时间。
category_id	否	Integer	分类ID。
tags	否	String	媒资标签。 单个标签不超过24个字节，最多不超过16个标签。 多个用逗号分隔，UTF8编码。
query_string	否	String	在媒资标题、描述中模糊查询的字符串。
page	否	Integer	分页编号，查询指定“asset_id”时，该参数无效。 若不设置或设置为null，默认查询首页，即第一页的数据，默认值：0。
size	否	Integer	每页记录数，查询指定“asset_id”时，该参数无效。 取值范围：[1,100]。 默认值：10。

请求参数

表 7-33 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

响应参数

状态码：200

表 7-34 响应 Body 参数

参数	参数类型	描述
asset_info_array	Array of AssetInfo objects	媒资信息列表。
is_truncated	Integer	列表是否被截断。 取值如下： <ul style="list-style-type: none"> • 1：表示本次查询未返回全部结果。 • 0：表示本次查询已经返回了全部结果。
total	Integer	查询媒资总数。>暂只能统计2万个媒资，若您需要查询具体的媒资总数，请提交工单人工申请。

表 7-35 AssetInfo

参数	参数类型	描述
asset_id	String	VOD分配的媒资ID，仅支持查询，不支持修改。
is_multi_transcode	Boolean	是否是多转码模式

参数	参数类型	描述
status	String	<p>媒资状态。</p> <p>取值如下：</p> <ul style="list-style-type: none"> • UNCREATED: 未创建（媒资ID不存在） • DELETED: 已删除 • CANCELLED: 上传取消 • SERVER_ERROR: 上传失败（点播服务端故障） • UPLOAD_FAILED: 上传失败（向OBS上传失败） • CREATING: 创建中 • PUBLISHED: 已发布 • WAITING_TRANSCODE: 待发布（转码排队中） • TRANSCODING: 待发布（转码中） • TRANSCODE_FAILED: 待发布（转码失败） • TRANSCODE_SUCCEED: 待发布（转码成功） • CREATED: 待发布（未转码） • NO_ASSET: 该媒资不存在 • DELETING: 正在删除 • DELETE_FAILED: 删除失败 • OBS_CREATING: OBS转存方式创建中 • OBS_CREATE_FAILED: OBS转存失败 • OBS_CREATE_SUCCESS: OBS转存成功
description	String	<p>媒资子状态或描述信息。</p> <ul style="list-style-type: none"> • 对于媒资异常场景，描述具体的异常原因。 • 对于正常场景，描述媒资的处理信息。
base_info	BaseInfo object	媒资基本信息。
play_info_array	Array of PlayInfo objects	<p>转码文件的播放信息。</p> <ul style="list-style-type: none"> • HLS或DASH: 此数组的成员个数为n+1, n为转码输出路数。 • MP4: 此数组的成员个数为n, n为转码输出路数。

表 7-36 BaseInfo

参数	参数类型	描述
title	String	媒资标题。 长度不超过128个字节，UTF8编码。
video_name	String	媒资文件名。
description	String	媒资描述。 长度不超过1024个字节。
category_id	Long	媒资分类id。
category_name	String	媒资分类名称。
create_time	String	媒资创建时间。 格式为yyyymmddhhmmss。必须是与时区无关的UTC时间。
last_modified	String	媒资最近修改时间。 格式为yyyymmddhhmmss。必须是与时区无关的UTC时间。
video_type	String	音视频文件类型。 取值如下： <ul style="list-style-type: none"> • 视频文件：MP4、TS、MOV、MXF、MPG、FLV、WMV、AVI、M4V、F4V、MPEG、3GP、ASF、MKV、WEBM • 音频文件：MP3、OGG、WAV、WMA、APE、FLAC、AAC、AC3、MMF、AMR、M4A、M4R、WV、MP2
tags	String	媒资标签。 单个标签不超过24个字节，最多不超过16个标签。 多个用逗号分隔，UTF8编码。
meta_data	MetaData object	视频的元数据信息。 经过视频解析后产生，包括封装格式、大小、分辨率、码率、帧率。
video_url	String	原始视频文件的访问地址。
sign_url	String	原视频文件的OBS临时访问地址。仅调用 查询指定媒资的详细信息 接口时才返回具体值。

参数	参数类型	描述
cover_info_array	Array of CoverInfo objects	封面信息。
subtitle_info	Array of SubtitleInfo objects	字幕信息数组
source_path	File_addr object	媒资存储参数信息。
output_path	File_addr object	媒资存储参数信息。

表 7-37 CoverInfo

参数	参数类型	描述
cover_url	String	封面文件的下载地址。

表 7-38 SubtitleInfo

参数	参数类型	描述
url	String	字幕文件的下载地址
id	Integer	字幕文件id
type	String	字幕文件类型
language	String	字幕文件语言种类

表 7-39 File_addr

参数	参数类型	描述
bucket	String	OBS的bucket名称。
location	String	桶所在的区域名，如“华北-北京四”的区域名为“cn-north-4”，创建的桶所在区域必须和点播服务所在区域保持一致。
object	String	文件的存储路径。

表 7-40 PlayInfo

参数	参数类型	描述
play_type	String	播放协议类型。 取值如下： <ul style="list-style-type: none"> • hls • dash • mp4
group_id	String	所属转码组Id
group_name	String	所属转码组名称
url	String	播放URL。
encrypted	Integer	标记流是否已被加密。 取值如下： <ul style="list-style-type: none"> • 0：表示未加密。 • 1：表示已被加密。 默认值：0。
meta_data	MetaData object	视频的元数据信息。 经过视频解析后产生，包括封装格式、大小、分辨率、码率、帧率。

表 7-41 MetaData

参数	参数类型	描述
pack_type	String	<p>视频封装格式。</p> <p>取值如下：</p> <ul style="list-style-type: none"> • MP4 • TS • MOV • MXF • MPG • FLV • WMV • MP3 • WMA • APE • FLAC • AAC • AC3 • MMF • AMR • M4A • M4R • OGG • WAV • WV • MP2 • AVI • F4V • M4V • MPEG • HLS • DASH

参数	参数类型	描述
codec	String	<p>视频编码格式。</p> <p>取值如下：</p> <ul style="list-style-type: none"> • MPEG-2 • MPEG-4 • H.264 • H.265 • WMV • Vorbis • AAC • AC-3 • AMR • APE • FLAC • MP3 • MP2 • WMA • PCM • ADPCM • WavPack <p>说明 如果codec返回unknown，说明不支持解析用户当前发送的音视频编码格式。</p>
duration	Long	<p>视频时长，单位：秒。</p> <p>若视频的原时长为非整数，则该字段值为原时长的向下取整。</p> <p>若视频的原时长小于1，则该字段值为1。</p>
duration_ms	Long	视频时长，单位毫秒。
video_size	Long	<p>视频文件大小。</p> <p>单位：字节。</p>
width	Long	<p>视频宽度（单位：像素）。</p> <ul style="list-style-type: none"> • 编码为H.264的取值范围：[32,3840]之间2的倍数。 • 编码为H.265的取值范围：[320,3840]之间4的倍数。

参数	参数类型	描述
hight	Long	视频高度（单位：像素）。 <ul style="list-style-type: none"> 编码为H.264的取值范围：[32,2160]之间2的倍数。 编码为H.265的取值范围：[240,2160]之间4的倍数。
height	Long	视频高度（单位：像素）
bit_rate	Long	视频平均码率。
frame_rate	Long	帧率（单位：帧每秒）。 取值如下： <ul style="list-style-type: none"> FRAMERATE_AUTO = 1, FRAMERATE_10 = 2, FRAMERATE_15 = 3, FRAMERATE_2397 = 4, // 23.97 fps FRAMERATE_24 = 5, FRAMERATE_25 = 6, FRAMERATE_2997 = 7, // 29.97 fps FRAMERATE_30 = 8, FRAMERATE_50 = 9, FRAMERATE_60 = 10 默认值：1。 单位：帧每秒。
quality	String	清晰度。 取值如下： <ul style="list-style-type: none"> FULL_HD：超高清 HD：高清 SD：标清 FLUENT：流畅 AD：自适应 2K 4K
audio_channels	Integer	音频的声道数。

状态码：400

表 7-42 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

- 单媒资ID查询
GET /v1.0/{project_id}/asset/info?asset_id=652c1e4085afeb22fdc256c6757d751b
- 多媒资ID查询
GET /v1.0/{project_id}/asset/info?asset_id=652c1e4085afeb22fdc256c6757d751b&asset_id=xxxxxx&asset_id=xxxxxx
- 按时间段查询
GET /v1.0/{project_id}/asset/info?start_time=20170725181000&end_time=20170726181000&page=0&size=20
- 查询（当前租户下）所有媒资：最近创建的10个，page为默认值0，size为默认值10
GET /v1.0/{project_id}/asset/info
- 查询（当前租户下）所有媒资：最近创建的第21个到第40个
GET /v1.0/{project_id}/asset/info?page=1&size=20
- 按分类查询
GET /v1.0/{project_id}/asset/info?category_id=100

响应示例

状态码：200

处理成功返回。

```
{
  "asset_info_array": [ {
    "asset_id": "652c1e4085afeb22fdc256c6757d751b",
    "status": "PUBLISHED",
    "description": "Asset meta is published",
    "is_multi_transcode": false,
    "base_info": {
      "title": "okFLV.flv",
      "video_name": "okFLV.flv",
      "description": null,
      "category_id": -1,
      "category_name": "其他",
      "create_time": "20190704144303",
      "last_modified": "20190704144303",
      "video_type": "FLV",
      "tags": null,
      "meta_data": {
        "pack_type": null,
        "codec": "H.264",
        "duration": 244,
        "duration_ms": 244000,
        "video_size": 13682041,
        "width": 512,
        "hight": 288,

```

```

    "height" : 288,
    "bit_rate" : 448,
    "frame_rate" : 30,
    "quality" : null
  },
  "video_url" : "https://651.cdn-vod.huaweicloud.com/asset/652c1e4085afeb22fdc256c6757d751b/45c2493abe68de3dac7e98b0dadcf8ce.flv",
  "sign_url" : "",
  "cover_info_array" : [ {
    "cover_url" : "https://651.cdn-vod.huaweicloud.com/asset/652c1e4085afeb22fdc256c6757d751b/cover/Cover0.jpg"
  } ],
  "subtitle_info" : [ ],
  "source_path" : {
    "bucket" : "obs-host",
    "location" : "cn-north-4",
    "object" : "1/okFLV.flv"
  },
  "output_path" : {
    "bucket" : "obs-host",
    "location" : "cn-north-4",
    "object" : "output/652c1e4085afeb22fdc256c6757d751b/"
  }
},
"play_info_array" : [ {
  "play_type" : "hls",
  "url" : "https://651.cdn-vod.huaweicloud.com/asset/652c1e4085afeb22fdc256c6757d751b/play_video/index.m3u8",
  "encrypted" : 0,
  "meta_data" : {
    "pack_type" : null,
    "codec" : "H.264",
    "duration" : 0,
    "duration_ms" : 0,
    "video_size" : 0,
    "width" : 0,
    "height" : 0,
    "bit_rate" : 0,
    "frame_rate" : 0,
    "quality" : null
  }
}, {
  "play_type" : "hls",
  "url" : "https://651.cdn-vod.huaweicloud.com/asset/652c1e4085afeb22fdc256c6757d751b/play_video/b5d498082bbcff7a2055041d803ae2f9_1.m3u8",
  "encrypted" : 0,
  "meta_data" : {
    "pack_type" : null,
    "codec" : "H.264",
    "duration" : 205,
    "duration_ms" : 205000,
    "video_size" : 14669824,
    "width" : 512,
    "height" : 288,
    "bit_rate" : 534,
    "frame_rate" : 0,
    "quality" : null
  }
}
]
},
"is_truncated" : 0,
"total" : 1
}

```

状态码：400

处理失败返回。

```
{
  "error_code": "VOD.10053",
  "error_msg": "The request parameter is illegal, illegal field: {xx}."
}
```

状态码

状态码	描述
200	处理成功返回。
400	处理失败返回。

错误码

请参见[错误码](#)。

7.5 修改媒资属性

功能介绍

修改媒资属性。

URI

PUT /v1.0/{project_id}/asset/info

表 7-43 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

请求参数

表 7-44 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。

参数	是否必选	参数类型	描述
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

表 7-45 请求 Body 参数

参数	是否必选	参数类型	描述
asset_id	是	String	VOD分配的媒资ID，仅支持查询，不支持修改。
title	否	String	媒资标题，长度不超过128个字节，UTF-8编码。
description	否	String	媒资描述，长度不超过1024个字节。
category_id	否	Integer	媒资分类id。
tags	否	String	媒资标签。 单个标签不超过24个字节，最多不超过16个标签。 多个用逗号分隔，UTF-8编码。

响应参数

状态码：400

表 7-46 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

修改媒资属性

```
PUT https://{endpoint}/v1.0/{project_id}/asset/info
```

```
Content-Type: application/json
```

```
{
  "asset_id": "f488337c31c8e4622f1590735b134c65",
  "title": "Avatar",
  "description": "Avatar, marketed as James Cameron's Avatar, is a 2009 American epic\ncscience fiction film directed, written, produced, and co-edited by\nJames Cameron",
  "category_id": "1"
}
```

响应示例

状态码：400

处理失败返回。

```
{
  "error_code": "VOD.10062",
  "error_msg": "Media asset or resource does not exist, please check."
}
```

状态码

状态码	描述
204	处理成功返回。
400	处理失败返回。

错误码

请参见[错误码](#)。

7.6 查询指定媒资的详细信息

功能介绍

查询指定媒资的详细信息。

URI

GET /v1.0/{project_id}/asset/details

表 7-47 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

表 7-48 Query 参数

参数	是否必选	参数类型	描述
asset_id	是	String	VOD分配的媒资ID，仅支持查询，不支持修改。

参数	是否必选	参数类型	描述
categories	否	Array of strings	<p>查询的信息类型。</p> <p>为空时表示查询所有信息。</p> <p>不为空时支持同时查询一个或者多个类型的信息，取值如下：</p> <ul style="list-style-type: none"> • base_info: 媒资基本信息。 • transcode_info: 转码结果信息。 • thumbnail_info: 截图结果信息。 • review_info: 审核结果信息。

请求参数

表 7-49 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	<p>用户Token，使用Token鉴权方式时必选。</p> <p>通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。</p>
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

响应参数

状态码：200

表 7-50 响应 Body 参数

参数	参数类型	描述
asset_id	String	VOD分配的媒资ID，仅支持查询，不支持修改。
base_info	BaseInfo object	媒资基本信息。

参数	参数类型	描述
transcode_info	TranscodeInfo object	转码生成文件信息。 说明 仅当转码成功后才能查询到此信息，未转码、正在转码以及转码失败时，无此字段信息。
thumbnail_info	ThumbnailInfo object	截图信息。 说明 仅当截图成功后才能查询到此信息，未截图、正在截图以及截图失败时，无此字段信息。
review_info	ReviewInfo object	审核信息数组。 说明 仅当审核成功后才能查询到此信息，未审核、正在审核以及审核失败时，无此字段信息。

表 7-51 BaseInfo

参数	参数类型	描述
title	String	媒资标题。 长度不超过128个字节，UTF8编码。
video_name	String	媒资文件名。
description	String	媒资描述。 长度不超过1024个字节。
category_id	Long	媒资分类id。
category_name	String	媒资分类名称。
create_time	String	媒资创建时间。 格式为yyyymmddhhmmss。必须是与时区无关的UTC时间。
last_modified	String	媒资最近修改时间。 格式为yyyymmddhhmmss。必须是与时区无关的UTC时间。

参数	参数类型	描述
video_type	String	音视频文件类型。 取值如下： <ul style="list-style-type: none"> • 视频文件：MP4、TS、MOV、MXF、MPG、FLV、WMV、AVI、M4V、F4V、MPEG、3GP、ASF、MKV、WEBM • 音频文件：MP3、OGG、WAV、WMA、APE、FLAC、AAC、AC3、MMF、AMR、M4A、M4R、WV、MP2
tags	String	媒资标签。 单个标签不超过24个字节，最多不超过16个标签。 多个用逗号分隔，UTF8编码。
meta_data	MetaData object	视频的元数据信息。 经过视频解析后产生，包括封装格式、大小、分辨率、码率、帧率。
video_url	String	原始视频文件的访问地址。
sign_url	String	原视频文件的OBS临时访问地址。仅调用 查询指定媒资的详细信息 接口时才返回具体值。
cover_info_array	Array of CoverInfo objects	封面信息。
subtitle_info	Array of SubtitleInfo objects	字幕信息数组
source_path	File_addr object	媒资存储参数信息。
output_path	File_addr object	媒资存储参数信息。

表 7-52 CoverInfo

参数	参数类型	描述
cover_url	String	封面文件的下载地址。

表 7-53 SubtitleInfo

参数	参数类型	描述
url	String	字幕文件的下载地址

参数	参数类型	描述
id	Integer	字幕文件id
type	String	字幕文件类型
language	String	字幕文件语言种类

表 7-54 File_addr

参数	参数类型	描述
bucket	String	OBS的bucket名称。
location	String	桶所在的区域名，如“华北-北京四”的区域名为“cn-north-4”，创建的桶所在区域必须和点播服务所在区域保持一致。
object	String	文件的存储路径。

表 7-55 TranscodeInfo

参数	参数类型	描述
template_group_name	String	转码模板组名称。
output	Array of Output objects	转码输出数组。 <ul style="list-style-type: none"> HLS或DASH格式：此数组的成员个数为n+1，n为转码输出路数。 MP4格式：此数组的成员个数为n，n为转码输出路数。
exec_desc	String	执行情况描述。
transcode_status	String	转码状态。 取值如下： <ul style="list-style-type: none"> UN_TRANSCODE：未转码 WAITING_TRANSCODE：待转码 TRANSCODING：转码中 TRANSCODE_SUCCEED：转码成功 TRANSCODE_FAILED：转码失败

表 7-56 Output

参数	参数类型	描述
play_type	String	协议类型。 取值如下： <ul style="list-style-type: none"> • hls • dash • mp4
url	String	播放URL。
group_id	String	所属转码组Id
group_name	String	所属转码组名称
encrypted	Integer	标记流是否已被加密。 取值如下： <ul style="list-style-type: none"> • 0：表示未加密。 • 1：表示已被加密。
quality	String	清晰度。 取值如下： <ul style="list-style-type: none"> • FLUENT：流畅 • SD：标清 • HD：高清 • FULL_HD：超清
meta_data	MetaData object	视频的元数据信息。 经过视频解析后产生，包括封装格式、大小、分辨率、码率、帧率。

表 7-57 MetaData

参数	参数类型	描述
pack_type	String	<p>视频封装格式。</p> <p>取值如下：</p> <ul style="list-style-type: none"> • MP4 • TS • MOV • MXF • MPG • FLV • WMV • MP3 • WMA • APE • FLAC • AAC • AC3 • MMF • AMR • M4A • M4R • OGG • WAV • WV • MP2 • AVI • F4V • M4V • MPEG • HLS • DASH

参数	参数类型	描述
codec	String	<p>视频编码格式。</p> <p>取值如下：</p> <ul style="list-style-type: none"> • MPEG-2 • MPEG-4 • H.264 • H.265 • WMV • Vorbis • AAC • AC-3 • AMR • APE • FLAC • MP3 • MP2 • WMA • PCM • ADPCM • WavPack <p>说明 如果codec返回unknown，说明不支持解析用户当前发送的音视频编码格式。</p>
duration	Long	<p>视频时长，单位：秒。</p> <p>若视频的原时长为非整数，则该字段值为原时长的向下取整。</p> <p>若视频的原时长小于1，则该字段值为1。</p>
duration_ms	Long	视频时长，单位毫秒。
video_size	Long	<p>视频文件大小。</p> <p>单位：字节。</p>
width	Long	<p>视频宽度（单位：像素）。</p> <ul style="list-style-type: none"> • 编码为H.264的取值范围：[32,3840]之间2的倍数。 • 编码为H.265的取值范围：[320,3840]之间4的倍数。

参数	参数类型	描述
hight	Long	视频高度（单位：像素）。 <ul style="list-style-type: none"> 编码为H.264的取值范围：[32,2160]之间2的倍数。 编码为H.265的取值范围：[240,2160]之间4的倍数。
height	Long	视频高度（单位：像素）
bit_rate	Long	视频平均码率。
frame_rate	Long	帧率（单位：帧每秒）。 取值如下： <ul style="list-style-type: none"> FRAMERATE_AUTO = 1, FRAMERATE_10 = 2, FRAMERATE_15 = 3, FRAMERATE_2397 = 4, // 23.97 fps FRAMERATE_24 = 5, FRAMERATE_25 = 6, FRAMERATE_2997 = 7, // 29.97 fps FRAMERATE_30 = 8, FRAMERATE_50 = 9, FRAMERATE_60 = 10 默认值：1。 单位：帧每秒。
quality	String	清晰度。 取值如下： <ul style="list-style-type: none"> FULL_HD：超高清 HD：高清 SD：标清 FLUENT：流畅 AD：自适应 2K 4K
audio_channels	Integer	音频的声道数。

表 7-58 ThumbnailInfo

参数	参数类型	描述
sample	Array of ThumbnailRsp objects	视频截图信息，截图类型为间隔。
dots	Array of ThumbnailRsp objects	视频截图信息，截图类型为时间点。
quantity	Array of ThumbnailRsp objects	视频截图信息，截图类型为数量。
exec_desc	String	执行情况描述。
thumbnail_status	String	截图状态。 取值如下： <ul style="list-style-type: none"> • UN_THUMBNAIL：未截图 • THUMBNAILING：截图中 • THUMBNAIL_SUCCEED：截图成功 • THUMBNAIL_FAILED：截图失败

表 7-59 ThumbnailRsp

参数	参数类型	描述
offset	Integer	截图在视频中的时间偏移，单位为秒。
url	String	截图访问URL

表 7-60 ReviewInfo

参数	参数类型	描述
suggestion	String	检测结果是否通过。 取值如下： <ul style="list-style-type: none"> • block：包含敏感信息，不通过。 • pass：不包含敏感信息，通过。 • review：需要人工复检。 说明 当同时检测多个场景时，suggestion的值以最可能包含敏感信息的场景为准。即任一场景出现了block则总的suggestion为block，所有场景都pass时suggestion为pass，这两种情况之外则一定有场景需要review，此时suggestion为review。

参数	参数类型	描述
text	TextReviewRet object	文本检测结果
cover	Array of PictureReviewRet objects	封面检测结果。
video	Array of PictureReviewRet objects	视频检测结果。
exec_desc	String	执行情况描述。
review_status	String	审核状态。 取值如下： <ul style="list-style-type: none"> • UN_REVIEW: 未审核 • REVIEWING: 审核中 • REVIEW_SUSPICIOUS: 审核可疑，需要人工审核 • REVIEW_PASSED: 审核通过 • REVIEW_FAILED: 审核失败。 • REVIEW_BLOCKED: 已屏蔽。

表 7-61 TextReviewRet

参数	参数类型	描述
suggestion	String	检测结果是否通过。 取值如下： <ul style="list-style-type: none"> • block: 包含敏感信息，不通过。 • pass: 不包含敏感信息，通过。 • review: 需要人工复检。
politics	String	涉政敏感词列表
porn	String	涉黄敏感词列表
abuse	String	辱骂敏感词列表

表 7-62 PictureReviewRet

参数	参数类型	描述
suggestion	String	检测结果是否通过。 取值如下： <ul style="list-style-type: none"> • block: 包含敏感信息，不通过。 • pass: 不包含敏感信息，通过。 • review: 需要人工复检。
offset	Integer	截图在视频中的时间偏移值。封面不涉及此字段 单位：秒。
url	String	对应截图/封面的访问URL。
politics	Array of ReviewDetail objects	政治因素审核结果。
terrorism	Array of ReviewDetail objects	暴恐元素审核结果。
porn	Array of ReviewDetail objects	涉黄内容审核结果。

表 7-63 ReviewDetail

参数	参数类型	描述
confidence	String	置信度。 取值范围：[0,1]。
label	String	每个检测结果的标签化说明。 <ul style="list-style-type: none"> • politics场景：label为对应的政治人物信息。 • terrorism场景：label为对应的暴恐元素（枪支、刀具、火灾等）信息。 • porn场景：label为对应的涉黄元素（涉黄、性感等）信息。

状态码：400

表 7-64 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

查询指定媒资详情

```
GET https://{endpoint}/v1.0/{project_id}/asset/details?asset_id={asset_id}
```

响应示例

状态码：200

处理成功返回。

```
{
  "asset_id": "41cff636d5b12a51e7eb2838bbf45201",
  "base_info": {
    "title": "Avatar",
    "video_name": "视频点播 VOD 视频.mp4",
    "description": "Avatar, marketed as James Cameron's Avatar, is a 2009 American\nepic science fiction
film directed, written, produced, and\nco-edited by James Cameron",
    "category_id": -1,
    "category_name": "其他",
    "create_time": "20190612032250",
    "last_modified": "20190613075030",
    "video_type": "MP4",
    "tags": "test1,test2",
    "meta_data": {
      "pack_type": null,
      "codec": "H.264",
      "duration": 131,
      "duration_ms": 131000,
      "video_size": 4942645,
      "width": 1280,
      "height": 720,
      "bit_rate": 173,
      "frame_rate": 30
    },
    "video_url": "https://355.cdn-vod.huaweicloud.com/asset/
41cff636d5b12a51e7eb2838bbf45201/5597e59de70722eaeb9b18c274e249b2.mp4",
    "sign_url": "https://vod-bucket-57-cn-north-4.obs.cn-
north-4.myhuaweicloud.com:443/05ab5cef408026f22f62c018de60cf2e/
41cff636d5b12a51e7eb2838bbf45201 /5597e59de70722eaeb9b18c274e249b2.mp4?
AWSAccessKeyId=CBN2J*****0RCSN&Expires=1652499973&Signature=kZYh0hEos2V*****AHGyXA
%3D",
    "cover_info_array": [ {
      "cover_url": "https://355.cdn-vod.huaweicloud.com/asset/41cff636d5b12a51e7eb2838bbf45201/cover/
Cover0.jpg"
    } ]
  },
  "transcode_info": {
    "template_group_name": "system_template_group",
    "output": [ {
      "play_type": "HLS",
      "url": "https://355.cdn-vod.huaweicloud.com/asset/41cff636d5b12a51e7eb2838bbf45201/play_video/
index.m3u8",
      "encrypted": 0,

```

```

"meta_data" : {
  "pack_type" : null,
  "codec" : "H.264",
  "duration" : 0,
  "duration_ms" : 0,
  "video_size" : 0,
  "width" : 0,
  "hight" : 0,
  "height" : 0,
  "bit_rate" : 0,
  "frame_rate" : 0
}
}, {
  "play_type" : "HLS",
  "url" : "https://355.cdn-vod.huaweicloud.com/asset/41cff636d5b12a51e7eb2838bbf45201/play_video/视
频点播VOD 视频_1_854X480_600_0.m3u8",
  "encrypted" : 0,
  "quality" : "SD",
  "meta_data" : {
    "pack_type" : null,
    "codec" : "H.264",
    "duration" : 130,
    "duration_ms" : 130000,
    "video_size" : 7976960,
    "width" : 854,
    "hight" : 480,
    "height" : 480,
    "bit_rate" : 421,
    "frame_rate" : 30000,
    "quality" : "SD"
  }
}, {
  "play_type" : "DASH",
  "url" : "https://355.cdn-vod.huaweicloud.com/asset/41cff636d5b12a51e7eb2838bbf45201/play_video/
index.mpd",
  "encrypted" : 0,
  "meta_data" : {
    "pack_type" : null,
    "codec" : "H.264",
    "duration" : 0,
    "duration_ms" : 0,
    "video_size" : 0,
    "width" : 0,
    "hight" : 0,
    "height" : 0,
    "bit_rate" : 0,
    "frame_rate" : 0
  }
}, {
  "play_type" : "DASH",
  "encrypted" : 0,
  "quality" : "SD",
  "meta_data" : {
    "pack_type" : null,
    "codec" : "H.264",
    "duration" : 130,
    "duration_ms" : 130000,
    "video_size" : 7976960,
    "width" : 854,
    "hight" : 480,
    "height" : 480,
    "bit_rate" : 421,
    "frame_rate" : 30000,
    "quality" : "SD"
  }
}
}],
"exec_desc" : "Transcode success",
"transcode_status" : "TRANSCODE_SUCCEED"

```

```
}  
}
```

状态码：400

处理失败返回。

```
{  
  "error_code" : "VOD.10062",  
  "error_msg" : "Media asset or resource does not exist, please check."  
}
```

状态码

状态码	描述
200	处理成功返回。
400	处理失败返回。

错误码

请参见[错误码](#)。

7.7 查询媒资列表

功能介绍

查询媒资列表，列表中的每一条记录包含媒资的概要信息。

URI

GET /v1.0/{project_id}/asset/list

表 7-65 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

表 7-66 Query 参数

参数	是否必选	参数类型	描述
asset_id	否	Array of strings	媒资ID，最多同时查询10个。

参数	是否必选	参数类型	描述
status	否	Array of strings	媒资状态，同时查询多个状态的媒资。 取值如下： <ul style="list-style-type: none"> • CREATING：上传中 • FAILED：上传失败 • CREATED：上传成功 • PUBLISHED：已发布 • TRANSCODING：转码中 • TRANSCODE_SUCCEED：转码成功 • TRANSCODE_FAILED：转码失败 • THUMBNAILING：截图中 • THUMBNAIL_SUCCEED：截图成功 • THUMBNAIL_FAILED：截图失败 • UN_REVIEW：未审核 • REVIEWING：审核中 • REVIEW_SUSPICIOUS：审核不过，待人工复审 • REVIEW_PASSED：审核通过 • REVIEW_FAILED：审核任务失败 • REVIEW_BLOCKED：已屏蔽
start_time	否	String	起始时间。 格式为yyyymmddhhmmss。必须是与时区无关的UTC时间。
end_time	否	String	结束时间。 格式为yyyymmddhhmmss。必须是与时区无关的UTC时间。
category_id	否	Integer	分类ID。
tags	否	String	媒资标签。单个标签不超过24个字节，最多不超过16个标签。多个用英文逗号分隔，UTF8编码。
query_string	否	String	在媒资标题、描述、分类名称中模糊查询的字符串。

参数	是否必选	参数类型	描述
media_type	否	Array of strings	音视频文件的格式，支持多格式查询，最多不超过20个。 取值如下： <ul style="list-style-type: none"> • 视频文件格式：MP4、TS、MOV、MXF、MPG、FLV、WMV、AVI、M4V、F4V、MPEG、WEBM • 音频文件格式：MP3、OGG、WAV、WMA、APE、FLAC、AAC、AC3、MMF、AMR、M4A、M4R、WV、MP2
page	否	Integer	分页编号。 默认值：0。
size	否	Integer	每页记录数。 取值范围：[1,100]。 默认值：10。

请求参数

表 7-67 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

响应参数

状态码：200

表 7-68 响应 Body 参数

参数	参数类型	描述
total	Integer	媒资总数 说明 暂只能统计2万个媒资，若您需要查询具体的媒资总数，请提交工单申请。
assets	Array of AssetSummary objects	媒资列表

表 7-69 AssetSummary

参数	参数类型	描述
asset_id	String	VOD分配的媒资ID，仅支持查询，不支持修改。
title	String	媒资标题。长度不超过128个字节，UTF-8编码。
description	String	媒资描述。长度不超过1024个字节。
duration	Integer	媒资时长。 单位：秒。
duration_ms	Long	视频时长，单位毫秒。
size	Long	媒资大小。 单位：字节。
original_url	String	原始播放url。
category	String	媒资分类名称。
covers	Array of CoverInfo objects	封面信息。
create_time	String	媒资创建时间。 格式为yyyymmddhhmmss。必须是与时区无关的UTC时间。
asset_status	String	媒资状态。 取值如下： <ul style="list-style-type: none"> CREATING：上传中。 FAILED：上传失败。 CREATED：上传成功。 PUBLISHED：已发布。 DELETED：已删除。

参数	参数类型	描述
transcode_status	String	转码状态。 取值如下： <ul style="list-style-type: none"> UN_TRANSCODE：未转码。 WAITING_TRANSCODE：待转码 TRANSCODING：转码中。 TRANSCODE_SUCCEED：转码成功。 TRANSCODE_FAILED：转码失败。
thumbnail_status	String	截图状态。 取值如下： <ul style="list-style-type: none"> UN_THUMBNAIL：未截图。 THUMBNAILING：截图中。 THUMBNAIL_SUCCEED：截图成功。 THUMBNAIL_FAILED：截图失败。
review_status	String	内容审核状态 取值如下： <ul style="list-style-type: none"> UN_REVIEW：未审核。 REVIEWING：审核中。 REVIEW_SUSPICIOUS：审核可疑，需要人工复审。 REVIEW_PASSED：审核通过。 REVIEW_FAILED：审核失败。 REVIEW_BLOCKED：已屏蔽。
exec_desc	String	媒资的任务执行描述汇总。 示例： <ul style="list-style-type: none"> asset_exec_desc: upload success, 媒资任务执行描述信息。 transcode_exec_desc: transcode success, 转码任务执行描述信息。 thumbnail_exec_desc: thumbnail failed, 截图任务执行描述信息。 review_exec_desc: review pass, 审核任务执行描述信息。

参数	参数类型	描述
media_type	String	音视频文件的格式。 取值如下： <ul style="list-style-type: none"> • 视频文件格式：MP4、TS、MOV、MXF、MPG、FLV、WMV、AVI、M4V、F4V、MPEG、WEBM • 音频文件格式：MP3、OGG、WAV、WMA、APE、FLAC、AAC、AC3、MMF、AMR、M4A、M4R、WV、MP2

表 7-70 CoverInfo

参数	参数类型	描述
cover_url	String	封面文件的下载地址。

状态码：400

表 7-71 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

查询媒资列表

GET https://{endpoint}/v1.0/{project_id}/asset/list

响应示例

状态码：200

处理成功返回。

```
{
  "total": 1,
  "assets": [ {
    "asset_id": "67d1470893419bfc9663103dd8a66ac",
    "title": "video.mp4",
    "duration": 60,
    "duration_ms": 60000,
    "size": 12881945,
    "category": "其他",
    "covers": [ {
      "cover_url": "https://355.cdn-vod.huaweicloud.com/asset/67d1470893419bfc9663103dd8a66ac/cover/
```

```
Cover0.jpg"
  }],
  "create_time": "20190625020756",
  "asset_status": "PUBLISHED",
  "transcode_status": "TRANSCODE_SUCCEED",
  "thumbnail_status": "UN_THUMBNAIL",
  "review_status": "UN_REVIEW",
  "exec_desc": "asset_exec_desc:Asset meta is\npublished;transcode_exec_desc:Transcode success;",
  "media_type": "MP4"
}]
}
```

状态码：400

处理失败返回。

```
{
  "error_code": "VOD.10053",
  "error_msg": "The request parameter is illegal, illegal field: {xx}."
}
```

状态码

状态码	描述
200	处理成功返回。
400	处理失败返回。

错误码

请参见[错误码](#)。

8 媒资预热

8.1 CDN 预热

功能介绍

媒资发布后，可通过指定媒资ID或URL向CDN预热。用户初次请求时，将由CDN节点提供请求媒资，加快用户下载缓存时间，提高用户体验。单租户每天最多预热1000个。

URI

POST /v1.0/{project_id}/asset/preheating

表 8-1 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

请求参数

表 8-2 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。

参数	是否必选	参数类型	描述
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

表 8-3 请求 Body 参数

参数	是否必选	参数类型	描述
asset_id	否	String	已发布媒资的ID。
urls	否	Array of strings	已发布媒资的播放URL列表，一次最多只能预热10个URL。

响应参数

状态码：202

表 8-4 响应 Body 参数

参数	参数类型	描述
task_id	String	预热任务ID。

状态码：400

表 8-5 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

- 预热媒资，按照媒资ID进行预热

```
POST https://{endpoint}/v1.0/{project_id}/asset/preheating
```

```
Content-Type: application/json
```

```
{
  "asset_id": "f488337c31c8e4622f1590735b134c65"
}
```

- 预热媒资，按照媒资URL进行预热

```
POST https://{endpoint}/v1.0/{project_id}/asset/preheating
```

```
Content-Type: application/json{ "urls": [ " https://example.com/asset/9db42f5e08c15edecd99a98da241994a/313bfd52a75f95ff48e8bf02eca2ab20.flv", " https://
```

```
example.com/asset/  
9e455adb02295aa123809e8dc7ca51c1/68b1241af3bf58bcde9914626e07f5af.mp4", " https://  
example.com/asset/9e455adb02295aa123809e8dc7ca51c1/play_video/  
68b1241af3bf58bcde9914626e07f5af_H.264_480X270_HEAACV1_300.mp4" ]}
```

响应示例

状态码：202

处理成功返回。

```
{  
  "task_id": "5199337c31c8e4622f1590735b13a263"  
}
```

状态码：400

处理失败返回。

```
{  
  "error_code": "VOD.10053",  
  "error_msg": "The request parameter is illegal, illegal field: {xx}."  
}
```

状态码

状态码	描述
202	处理成功返回。
400	处理失败返回。

错误码

请参见[错误码](#)。

8.2 查询 CDN 预热

功能介绍

查询预热结果。

URI

GET /v1.0/{project_id}/asset/preheating

表 8-6 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

表 8-7 Query 参数

参数	是否必选	参数类型	描述
task_id	是	String	任务ID

请求参数

表 8-8 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

响应参数

状态码：200

表 8-9 响应 Body 参数

参数	参数类型	描述
preheating_results	Array of PreheatingResult objects	预热任务数组

表 8-10 PreheatingResult

参数	参数类型	描述
url	String	媒资URL。
status	String	预热任务状态。 取值如下： <ul style="list-style-type: none"> processing：处理中 succeed：预热完成 failed：预热失败

状态码：400

表 8-11 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

```
GET https://{endpoint}/v1.0/{project_id}/asset/preheating?task_id={task_id}
```

响应示例

状态码：200

处理成功返回。

```
{
  "preheating_results": [ {
    "url": "https://example.com/asset/9db42f5e08c15edecd99a98da241994a/
313bfd52a75f95ff48e8bf02eca2ab20.flv",
    "status": "processing"
  }, {
    "url": "https://example.com/asset/
9e455adb02295aa123809e8dc7ca51c1/68b1241af3bf58bcde9914626e07f5af.mp4",
    "status": "succeed"
  }, {
    "url": "https://example.com/asset/9e455adb02295aa123809e8dc7ca51c1/play_video/
68b1241af3bf58bcde9914626e07f5af_H.264_480X270_HEAACV1_300.mp4",
    "status": "failed"
  }
]
```

状态码：400

处理失败返回。

```
{
  "error_code": "VOD.10053",
  "error_msg": "The request parameter is illegal, illegal field: {xx}."
}
```

状态码

状态码	描述
200	处理成功返回。
400	处理失败返回。

错误码

请参见[错误码](#)。

9 媒资分类

9.1 创建媒资分类

功能介绍

创建媒资分类。

接口约束

最大支持三级分类，每个分类最多支持创建128个子分类。

URI

POST /v1.0/{project_id}/asset/category

表 9-1 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

请求参数

表 9-2 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

参数	是否必选	参数类型	描述
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

表 9-3 请求 Body 参数

参数	是否必选	参数类型	描述
name	是	String	媒资分类名称，最大64字节。
parent_id	否	Integer	父分类ID。 若不填，则默认生成一级分类。 根节点分类ID为0。

响应参数

状态码：200

表 9-4 响应 Body 参数

参数	参数类型	描述
name	String	媒资分类名称。
parent_id	Integer	父分类ID。 一级分类父ID为0。
id	Integer	媒资分类ID。
level	Integer	媒资分类层级。 取值如下： <ul style="list-style-type: none"> • 1：一级分类层级。 • 2：二级分类层级。 • 3：三级分类层级。
projectId	String	项目ID。

状态码：400

表 9-5 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

创建媒资分类

POST https://{endpoint}/v1.0/{project_id}/asset/category

Content-Type: application/json

```
{
  "name": "电影"
}
```

响应示例

状态码：200

处理成功返回。

```
{
  "id" : 334,
  "name" : "电影",
  "parent_id" : 0,
  "level" : 1,
  "projectId" : "58411d8df9064*****75d75b54e01358"
}
```

状态码：400

处理失败返回。

```
{
  "error_code" : "VOD.10053",
  "error_msg" : "The request parameter is illegal, illegal field: {xx}."
}
```

状态码

状态码	描述
200	处理成功返回。
400	处理失败返回。

错误码

请参见[错误码](#)。

9.2 修改媒资分类

功能介绍

修改媒资分类。

URI

PUT /v1.0/{project_id}/asset/category

表 9-6 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

请求参数

表 9-7 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

表 9-8 请求 Body 参数

参数	是否必选	参数类型	描述
name	是	String	媒资分类名称。
id	是	Integer	媒资分类ID。

响应参数

状态码：200

表 9-9 响应 Body 参数

参数	参数类型	描述
name	String	媒资分类名称。
parent_id	Integer	父分类ID。 一级分类父ID为0。
id	Integer	媒资分类ID。
level	Integer	媒资分类层级。 取值如下： <ul style="list-style-type: none"> • 1：一级分类层级。 • 2：二级分类层级。 • 3：三级分类层级。
projectId	String	项目ID。

状态码：400

表 9-10 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

修改媒资分类

```
PUT https://{endpoint}/v1.0/{project_id}/asset/category
```

```
Content-Type: application/json
```

```
{
  "id": 334,
  "name": "film"
}
```

响应示例

状态码：200

处理成功返回。

```
{
  "id" : 334,
  "name" : "film",
  "parent_id" : 0,
  "level" : 1,
  "projectId" : "58411d8df90649a39b75d75b54e01358"
}
```

状态码：400

处理失败返回。

```
{
  "error_code": "VOD.10053",
  "error_msg": "The request parameter is illegal, illegal field: {xx}."
}
```

状态码

状态码	描述
200	处理成功返回。
400	处理失败返回。

错误码

请参见[错误码](#)。

9.3 删除媒资分类

功能介绍

删除媒资分类。

URI

DELETE /v1.0/{project_id}/asset/category

表 9-11 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

表 9-12 Query 参数

参数	是否必选	参数类型	描述
id	是	Integer	视频分类ID。 若设置为0或者null，则删除项目id下所有一级分类。

请求参数

表 9-13 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

响应参数

状态码：400

表 9-14 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

删除媒资分类

```
DELETE https://{endpoint}/v1.0/{project_id}/asset/category?id={id}
```

响应示例

状态码：400

处理失败返回。

```
{
  "error_code" : "VOD.10053",
  "error_msg" : "The request parameter is illegal, illegal field: {xx}."
}
```

状态码

状态码	描述
204	处理成功返回204 No Content。

状态码	描述
400	处理失败返回。

错误码

请参见[错误码](#)。

9.4 查询指定分类信息

功能介绍

查询指定分类信息，及其子分类（即下一级分类）的列表。

URI

GET /v1.0/{project_id}/asset/category

表 9-15 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

表 9-16 Query 参数

参数	是否必选	参数类型	描述
id	是	Integer	视频分类ID。 若设置为0或者null，则查询项目Id下所有一级分类。

请求参数

表 9-17 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

参数	是否必选	参数类型	描述
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

响应参数

状态码：200

表 9-18 响应 Body 参数

参数	参数类型	描述
[数组元素]	Array of QueryCategoryRsp objects	分类返回值

表 9-19 QueryCategoryRsp

参数	参数类型	描述
id	String	分类ID。
name	String	分类名称。
children	Array of QueryCategoryRsp objects	子分类列表。

状态码：400

表 9-20 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

查询指定分类信息

```
GET https://{endpoint}/v1.0/{project_id}/asset/category?id={id}
```

响应示例

状态码：200

处理成功返回。

```
[{
  "id": "334",
  "name": "电影",
  "children": [{
    "id": "335",
    "name": "电视",
    "children": []
  }]
}]
```

状态码：400

处理失败返回。

```
{
  "error_code": "VOD.10053",
  "error_msg": "The request parameter is illegal, illegal field: {xx}."
}
```

状态码

状态码	描述
200	处理成功返回。
400	处理失败返回。

错误码

请参见[错误码](#)。

10 密钥查询

10.1 查询密钥

功能介绍

终端播放HLS加密视频时，向租户管理系统请求密钥，租户管理系统先查询其本地有没有已缓存的密钥，没有时则调用此接口向VOD查询。该接口的具体使用场景请参见《点播服务最佳实践》中通过HLS加密防止视频泄露的示例代码。

URI

GET /v1.0/{project_id}/asset/ciphers

表 10-1 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

表 10-2 Query 参数

参数	是否必选	参数类型	描述
asset_id	是	String	VOD分配的媒资ID，仅支持查询，不支持修改。

请求参数

表 10-3 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

响应参数

状态码：200

表 10-4 响应 Body 参数

参数	参数类型	描述
asset_id	String	VOD分配的媒资ID，仅支持查询，不支持修改。
edk	String	密钥密文。
dk	String	密钥明文。

状态码：400

表 10-5 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

查询密钥

```
GET https://{endpoint}/v1.0/{project_id}/asset/ciphers?asset_id={asset_id}
```

响应示例

状态码：200

处理成功返回。

```
{
  "asset_id": "06da0367af8f297ea3efc791e8b27157",
  "edk":
  "02009800f00446c3aa6ad610bb121a602fa3091daca7a544ded4f514cae2c233cf51f3d88de64712242a846804
  2db19b4282e0d7196a8d4df6150169debb5d077527fad983e14174f1ab5430958ca4903187321f0a3406284bf9
  38b64de95ddd3db438df9ddae98d347058969fd39be5049a0a5830396432303366352d313235362d346265322
  d626539642d3831316433336333663035320000000f9776c60d33706c68195bc64934f94ee28fbfd03e5fbb65
  c5450599d6aa4807",
  "dk": "4Q9KYgKbUChxocNYbteP3A=="
}
```

状态码：400

处理失败返回。

```
{
  "error_code": "VOD.10101",
  "error_msg": "Unable to get the key because there is no encryption."
}
```

状态码

状态码	描述
200	处理成功返回。
400	处理失败返回。

错误码

请参见[错误码](#)。

11 统计分析

11.1 查询 CDN 统计信息

功能介绍

查询CDN的统计数据，包括流量、峰值带宽、请求总数、请求命中率、流量命中率。查询存在1小时误差。

URI

GET /v1.0/{project_id}/asset/cdn-statistics

表 11-1 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

表 11-2 Query 参数

参数	是否必选	参数类型	描述
start_time	否	String	起始时间，格式为yyyymmddhhmmss。

参数	是否必选	参数类型	描述
end_time	否	String	<p>结束时间，格式为 yyyymmddhhmmss。</p> <ul style="list-style-type: none"> “start_time”、“end_time”均不存在时，“start_time”取当天零点，“end_time”取当前时间。 “start_time”不存在、“end_time”存在，请求非法。 “start_time”存在、“end_time”不存在，“end_time”取当前时间。 只能查询最近三个月内的数据，且时间跨度不能超过31天。 起始时间和结束时间会自动规整，起始时间规整为指定时间所在的整点时刻，结束时间规整为指定时间所在时间的下一小时整点时刻。
stat_type	是	String	<p>统计数据类型。 取值如下：</p> <ul style="list-style-type: none"> cdn_bw: CDN峰值带宽 cdn_flux: CDN流量 req_num: 请求总数 req_hit_rate: 请求命中率 flux_hit_rate: 流量命中率 <p>每次只能查询一种统计数据。</p>
domain	是	String	<p>域名列表，多个域名以逗号（半角）分隔。 示例： example.test1.com,example.test2.com。 ALL表示查询名下全部域名。一次最多查询100个域名。</p>

参数	是否必选	参数类型	描述
interval	否	Integer	<p>查询粒度间隔。</p> <p>取值如下：</p> <ul style="list-style-type: none"> 时间跨度1天：5分钟、1小时、4小时、8小时，分别对应300秒、3600秒、14400秒和28800秒。 时间跨度2~7天：1小时、4小时、8小时、1天，分别对应3600秒、14400秒、28800秒和86400秒。 时间跨度8~31天：4小时、8小时、1天，分别对应14400秒、28800秒和86400秒。 <p>单位：秒。</p> <p>若不设置，默认取对应时间跨度的最小间隔。</p>

请求参数

表 11-3 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	<p>用户Token，使用Token鉴权方式时必选。</p> <p>通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。</p>
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

响应参数

状态码：200

表 11-4 响应 Body 参数

参数	参数类型	描述
start_time	String	统计起始时间。

参数	参数类型	描述
interval	Integer	统计间隔。
values	Array of longs	采样数据数组。 从start_time开始，每个间隔对应一个采样数据（其中流量单位：Byte，带宽单位：bps）。

状态码：400

表 11-5 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

查询CDN统计数据

GET https://{endpoint}/v1.0/{project_id}/asset/cdn-statistics?domain=www.example.com&stat_type=cdn_bw

响应示例

状态码：200

处理成功返回。

```
{
  "start_time": "string",
  "interval": 0,
  "values": [ 0 ]
}
```

状态码：400

处理失败返回。

```
{
  "error_code": "VOD.10053",
  "error_msg": "The request parameter is illegal, illegal field: {xx}."
}
```

状态码

状态码	描述
200	处理成功返回。
400	处理失败返回。

错误码

请参见[错误码](#)。

11.2 查询源站统计信息

功能介绍

查询点播源站的统计数据，包括存储空间、转码时长。

源站统计数据仅支持查最近一个月的历史数据。

URI

GET /v1.0/{project_id}/asset/vod-statistics

表 11-6 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

表 11-7 Query 参数

参数	是否必选	参数类型	描述
start_time	否	String	起始时间，格式为 yyyymmddhhmmss。

参数	是否必选	参数类型	描述
end_time	否	String	<p>结束时间，格式为 yyyymmddhhmmss。</p> <ul style="list-style-type: none"> “start_time”、“end_time”均不存在时，“start_time”取当天零点，“end_time”取当前时间。 “start_time”不存在、“end_time”存在，请求非法。 “start_time”存在、“end_time”不存在，“end_time”取当前时间。 只能查询最近三个月内的数据，且时间跨度不能超过31天。 起始时间和结束时间会自动规整，起始时间规整为指定时间所在的整点时刻，结束时间规整为指定时间所在时间的下一小时整点时刻。
interval	否	Integer	<p>查询粒度间隔。</p> <p>取值如下：</p> <ul style="list-style-type: none"> 时间跨度1天：1小时、4小时、8小时，分别对应3600秒、14400秒和28800秒。 时间跨度2~7天：1小时、4小时、8小时、1天，分别对应3600秒、14400秒、28800秒和86400秒。 时间跨度8~31天：4小时、8小时、1天，分别对应14400秒、28800秒和86400秒。 <p>单位：秒。</p> <p>若不设置，默认取对应时间跨度的最小间隔。</p>

请求参数

表 11-8 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

响应参数

状态码：200

表 11-9 响应 Body 参数

参数	参数类型	描述
start_time	String	统计起始时间。
interval	Integer	统计间隔。
sample_data	Array of VodSampleData objects	采样数据数组。从start_time开始，每个间隔对应一个采样数据（其中转码时长单位：min，存储单位：GB）。

表 11-10 VodSampleData

参数	参数类型	描述
storage	Float	标准存储空间。 单位：GB。
storage_warm	Float	低频存储空间。 单位：GB。
storage_cold	Float	归档存储空间。 单位：GB。
transcode	Long	转码时长。 单位：分钟。

状态码：400

表 11-11 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

查询点播源站统计数据

GET https://{endpoint}/v1.0/{project_id}/asset/vod-statistics?start_time={start_time}

响应示例

状态码：200

处理成功返回。

```
{
  "start_time": "20190528000000",
  "interval": 3600,
  "sample_data": [ {
    "storage": 0,
    "transcode": 0
  } ]
}
```

状态码：400

处理失败返回。

```
{
  "error_code": "VOD.10053",
  "error_msg": "The request parameter is illegal, illegal field: {xx}."
}
```

状态码

状态码	描述
200	处理成功返回。
400	处理失败返回。

错误码

请参见[错误码](#)。

11.3 查询 TopN 媒资信息

功能介绍

查询指定域名在指定日期播放次数排名Top 100的媒资统计数据。

URI

GET /v1.0/{project_id}/asset/top-statistics

表 11-12 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

表 11-13 Query 参数

参数	是否必选	参数类型	描述
domain	是	String	查询域名，暂只支持查询单个或者全部域名。 取值如下： <ul style="list-style-type: none"> • 单个加速域名，格式：example.test1.com。 • ALL：表示查询名下全部域名。
date	是	String	查询日期，格式为yyyymmdd <ul style="list-style-type: none"> • date必须为昨天或之前的日期。 • 最多只能查最近一个月内的数据。

请求参数

表 11-14 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

响应参数

状态码：200

表 11-15 响应 Body 参数

参数	参数类型	描述
top_urls	Array of TopUrl objects	指定域名top100 URL集合

表 11-16 TopUrl

参数	参数类型	描述
value	Long	总播放次数。
asset_id	String	VOD分配的媒资ID，仅支持查询，不支持修改。
title	String	媒资名称。
duration	Integer	媒资时长。 单位：秒。
duration_ms	Long	视频时长，单位毫秒。
size	Long	媒资原始大小。 单位：字节。

状态码：400

表 11-17 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

查询top100

GET https://{endpoint}/v1.0/{project_id}/asset/top-statistics?domain=ALL&date=20190612

响应示例

状态码：200

处理成功返回。

```
{
  "top_urls": [ {
    "value": 0,
    "asset_id": "548b6023bf2cc925bd64343f04ca0f88",
    "title": "test",
    "duration": 0,
    "duration_ms": 0,
    "size": 0
  } ]
}
```

状态码：400

处理失败返回。

```
{
  "error_code": "VOD.10053",
  "error_msg": "The request parameter is illegal, illegal field: {xx}."
}
```

状态码

状态码	描述
200	处理成功返回。
400	处理失败返回。

错误码

请参见[错误码](#)。

11.4 查询域名播放日志

功能介绍

查询指定点播域名某段时间内在CDN的相关日志。

URI

GET /v1.0/{project_id}/vod/cdn/logs

表 11-18 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

表 11-19 Query 参数

参数	是否必选	参数类型	描述
domain_name	是	String	加速域名，参考格式： www.test1.com。
query_date	是	String	查询开始时间，格式为 yyyyMMddHHmmss。 <ul style="list-style-type: none"> 查询结果为开始时间之后24小时内的日志数据 只能查最近一个月内的数据
page_size	否	Integer	每页显示日志数量。取值范围为 [1,10000]，默认值：10000。
page_number	否	Integer	当前页数。取值范围为 [1,65535]，默认值：1。

请求参数

表 11-20 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

参数	是否必选	参数类型	描述
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

响应参数

状态码：200

表 11-21 响应 Body 参数

参数	参数类型	描述
total	Integer	日志总数。
logs	Array of CdnLog objects	日志列表数据。

表 11-22 CdnLog

参数	参数类型	描述
domain_name	String	域名名称。
start_time	String	查询起始时间。
end_time	String	查询结束时间。
name	String	日志名称。
size	Long	日志大小。 单位：byte。
link	String	日志下载链接。 日志文件中的字段说明请参见 日志管理 。

状态码：400

表 11-23 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

查询播放日志

```
GET https://{endpoint}/v1.0/{project_id}/vod/cdn/logs?domain_name=1866.cdn-vod.huaweicloud.com&query_date=20211223000000
```

响应示例

状态码：200

处理成功返回。

```
{
  "total": 1,
  "logs": [ {
    "domain_name": "1866.cdn-vod.huaweicloud.com",
    "start_time": "2021-12-23 23:00:00",
    "end_time": "2021-12-24 00:00:00",
    "name": "2021122323-1866.cdn-vod.huaweicloud.com-cn.gz",
    "size": 225,
    "link": "http://cdn-log-user-bj4.obs.cn-north-4.myhuaweicloud.com:80/mainland/20211223/23/2021122323-1866.cdn-vod.huaweicloud.com-cn.gz?AccessKeyId=AOV5GWALBMNTLIYDEQLV&Expires=1641433717&response-content-disposition=attachment%3Bfilename%3D%222021122323-1866.cdn-vod.huaweicloud.com-cn.gz%22&Signature=kZYh0hEos2V*****AHGyXA%3D"
  } ]
}
```

状态码：400

处理失败返回。

```
{
  "error_code": "VOD.10053",
  "error_msg": "The request parameter is illegal, illegal field: {xx}."
}
```

状态码

状态码	描述
200	处理成功返回。
400	处理失败返回。

错误码

请参见[错误码](#)。

11.5 查询媒资日播放统计数据

功能介绍

查询媒资日播放统计数据。

使用媒资日播放统计查询API前，需要先提交工单开通统计功能，才能触发统计任务。

支持查询最近一年的播放统计数据。

URI

GET /v1/{project_id}/asset/daily-summary

表 11-24 路径参数

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

表 11-25 Query 参数

参数	是否必选	参数类型	描述
start_time	是	String	查询开始时间。仅支持查询一年内的数据，且一次查询的日期跨度不能超过90天。 如果查询指定开始日期的数据，格式为：yyyyMMdd000000。
end_time	是	String	查询结束时间。仅支持查询一年内的数据，且一次查询的日期跨度不能超过90天。 如果查询指定结束日期的数据，格式为：yyyyMMdd000000。
offset	否	Integer	偏移量，表示查询该偏移量后面的记录。
limit	否	Integer	查询返回记录的数量限制。

请求参数

表 11-26 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

响应参数

状态码：200

表 11-27 响应 Body 参数

参数	参数类型	描述
total	Integer	记录总数
summary_results	Array of AssetDailySummaryResult objects	日志文件列表

表 11-28 AssetDailySummaryResult

参数	参数类型	描述
date	String	播放日期，格式为：yyyyMMdd000000。
link	String	<p>日播放统计数据文件的下载地址，有效期为12个小时。</p> <p>文件内容格式：[域名]\t[媒资ID]\t[日期]\t[播放流量]\t[播放次数]</p> <p>播放次数统计说明：</p> <ul style="list-style-type: none"> • HLS文件：访问M3U8文件时会统计播放次数，访问TS文件时不会统计播放次数。 • 其它文件：如MP4文件，当播放请求带有range且range的start参数不等于0时，不统计播放次数。其它情况下，会统计播放次数。

状态码：400

表 11-29 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

无

响应示例

状态码：200

处理成功返回。

```
{
  "total": 1,
  "summary_results": [ {
    "date": "20231201000000",
    "link": "http://cdn-log-user-bj4.obs.cn-north-4.myhuaweicloud.com:80/asset-play-summary/05a8aee14d8026a92fcdc018fe235c2c/20231201-asset-play-summary.gz?AccessKeyId=QXBNBGA2GW9EP2XXOYUN&Expires=1701786544&Signature=kZYh0hEos2V*****AHGyXA%3D"
  } ]
}
```

状态码：400

处理失败返回。

```
{
  "error_code": "VOD.10053",
  "error_msg": "The request parameter is illegal, illegal field: {xx}."
}
```

状态码

状态码	描述
200	处理成功返回。
400	处理失败返回。

错误码

请参见[错误码](#)。

12 水印模板管理

12.1 创建水印模板

功能介绍

创建水印模板。

URI

POST /v1.0/{project_id}/template/watermark

表 12-1 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

请求参数

表 12-2 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。

参数	是否必选	参数类型	描述
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

表 12-3 请求 Body 参数

参数	是否必选	参数类型	描述
name	是	String	水印模板名称。
watermark_type	否	String	水印类型，当前只支持IMAGE（图片水印）。
image_process	否	String	watermark_type设置为IMAGE时有效。 目前包括： <ul style="list-style-type: none"> ● ORIGINAL：只做简单缩放，不做其他处理 ● TRANSPARENT：图片底色透明 ● GRAYED：彩色图片变灰
dx	否	String	水印图片相对输出视频的水平偏移量，默认值是0。 设置方法有两种，建议小数型，整数型可能导致控制台图片无法预览，但不影响转码： <ul style="list-style-type: none"> ● 整数型：表示图片起点水平偏移视频顶点的像素值，单位px。取值范围：[0, 4096] ● 小数型：表示图片起点相对于视频分辨率宽的水平偏移比率。取值范围：(0, 1)，支持4位小数，如0.9999，超出部分系统自动丢弃。

参数	是否必选	参数类型	描述
dy	否	String	<p>水印图片相对输出视频的垂直偏移量，默认值是0。</p> <p>设置方法有两种，建议小数型，整数型可能导致控制台图片无法预览，但不影响转码：</p> <ul style="list-style-type: none"> ● 整数型：表示图片起点垂直偏移视频顶点的像素值，单位px。取值范围：[0, 4096] ● 小数型：表示图片起点相对于视频分辨率高的垂直偏移比率。取值范围：(0, 1)，支持4位小数，如0.9999，超出部分系统自动丢弃。
position	否	String	<p>水印的位置，默认TOPRIGHT。目前包括：</p> <ul style="list-style-type: none"> ● TOPRIGHT：右上位置 ● TOPLEFT：左上位置 ● BOTTOMRIGHT：右下位置 ● BOTTOMLEFT：左下位置
width	否	String	<p>水印图片宽，建议小数型，整数型可能导致控制台图片无法预览，但不影响转码。</p> <p>值有两种形式：</p> <ul style="list-style-type: none"> ● 整数型代表水印图片宽的像素值，范围[8, 4096]，单位px。 ● 小数型代表相对输出视频分辨率宽的比率，范围(0,1)，支持4位小数，如0.9999，超出部分系统自动丢弃。
height	否	String	<p>水印图片高，建议小数型，整数型可能导致控制台图片无法预览，但不影响转码。</p> <p>值有两种形式：</p> <ul style="list-style-type: none"> ● 整数型代表水印图片高的像素值，范围[8, 4096]，单位px。 ● 小数型代表相对输出视频分辨率高的比率，范围(0, 1)，支持4位小数，如0.9999，超出部分系统自动丢弃。

参数	是否必选	参数类型	描述
timeline_start	否	String	水印开始时间, 与 "timeline_duration"配合使用。 取值范围:[0, END)。"END"表示视频结束时间。 单位:秒。
timeline_duration	否	String	水印持续时间, 与 "timeline_start"配合使用。 取值范围:(0,END-开始时间]。 "END"表示视频结束时间。 单位:秒。 默认:END。
type	是	String	仅支持png、jpg和jpeg格式的图片作为水印模板, 传值仅支持JPG、JPEG和PNG三种字符串。
md5	否	String	水印图片MD5值。

响应参数

状态码: 200

表 12-4 响应 Body 参数

参数	参数类型	描述
id	String	水印配置模板id
upload_url	String	水印图片上传地址

状态码: 400

表 12-5 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

创建水印模板

```
POST https://{endpoint}/v1.0/{project_id}/template/watermark
```

```
Content-Type: application/json
{
  "name": "test",
  "type": "PNG"
}
```

响应示例

状态码：200

处理成功返回

```
{
  "id": "a52ba84366abebb4c4614e1b16973549",
  "upload_url": "https://vod-bucket-26-cn-north-4.obs.cn-
north-4.myhuaweicloud.com:443/05ab5cef408026f22f62c018de60cf2e/
a52ba84366abebb4c4614e1b16973549/watermark.png?
AWSAccessKeyId=CBN2J*****ORCSN&Expires=1625479312&Signature=kZYh0hEos2V*****AHGyXA
%3D"
}
```

状态码：400

处理失败返回

```
{
  "error_code": "VOD.10053",
  "error_msg": "The request parameter is illegal, illegal field: {xx}."
}
```

状态码

状态码	描述
200	处理成功返回
400	处理失败返回

错误码

请参见[错误码](#)。

12.2 修改水印模板

功能介绍

修改水印模板

URI

PUT /v1.0/{project_id}/template/watermark

表 12-6 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID, 获取方法请参考 获取项目ID

请求参数

表 12-7 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token, 使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取(响应消息头中X-Subject-Token的值)。
Authorization	否	String	使用AK/SK方式认证时必选, 携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选, 请求的发生时间。

表 12-8 请求 Body 参数

参数	是否必选	参数类型	描述
id	是	String	水印模板配置id
name	是	String	水印模板名称
dx	否	String	水印图片相对输出视频的水平偏移量, 默认值是0。 设置方法有两种, 建议小数型, 整数型可能导致控制台图片无法预览, 但不影响转码: <ul style="list-style-type: none"> 整数型: 表示图片起点水平偏移视频顶点的像素值, 单位px。取值范围: [0, 4096] 小数型: 表示图片起点相对于视频分辨率宽的水平偏移比率。取值范围: (0, 1), 支持4位小数, 如0.9999, 超出部分系统自动丢弃。

参数	是否必选	参数类型	描述
dy	否	String	<p>水印图片相对输出视频的垂直偏移量，默认值是0。</p> <p>设置方法有两种，建议小数型，整数型可能导致控制台图片无法预览，但不影响转码：</p> <ul style="list-style-type: none"> ● 整数型：表示图片起点垂直偏移视频顶点的像素值，单位px。取值范围：[0, 4096] ● 小数型：表示图片起点相对于视频分辨率高的垂直偏移比率。取值范围：(0, 1)，支持4位小数，如0.9999，超出部分系统自动丢弃。
position	否	String	<p>水印的位置，默认TOPRIGHT。目前包括：</p> <ul style="list-style-type: none"> ● TOPRIGHT：右上位置 ● TOPLEFT：左上位置 ● BOTTOMRIGHT：右下位置 ● BOTTOMLEFT：左下位置
width	否	String	<p>水印图片宽，建议小数型，整数型可能导致控制台图片无法预览，但不影响转码。</p> <p>值有两种形式：</p> <ul style="list-style-type: none"> ● 整数型代表水印图片宽的像素值，范围[8, 4096]，单位px。 ● 小数型代表相对输出视频分辨率宽的比率，范围(0,1)，支持4位小数，如0.9999，超出部分系统自动丢弃。
height	否	String	<p>水印图片高，建议小数型，整数型可能导致控制台图片无法预览，但不影响转码。</p> <p>值有两种形式：</p> <ul style="list-style-type: none"> ● 整数型代表水印图片高的像素值，范围[8, 4096]，单位px。 ● 小数型代表相对输出视频分辨率高的比率，范围(0, 1)，支持4位小数，如0.9999，超出部分系统自动丢弃。

参数	是否必选	参数类型	描述
watermark_type	否	String	水印类型，当前只支持IMAGE（图片水印）
image_processes	否	String	watermark_type设置为IMAGE时有效。 目前包括： <ul style="list-style-type: none"> • ORIGINAL：只做简单缩放，不做其他处理 • TRANSPARENT：图片底色透明 • GRAYED：彩色图片变灰
timeline_start	否	String	水印开始时间，与"timeline_duration"配合使用。 取值范围:[0, END)。"END"表示视频结束时间。 单位:秒。
timeline_duration	否	String	水印持续时间，与"timeline_start"配合使用。 取值范围:(0,END-开始时间]。 "END"表示视频结束时间。 单位:秒。 默认:END。

响应参数

状态码：400

表 12-9 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

修改水印模板

```
PUT https://{endpoint}/v1.0/{project_id}/template/watermark
Content-Type: application/json
{
  "id": "2305739f855413a84af9e6ad6ebb21be",
  "name": "test"
}
```

响应示例

状态码：400

处理失败返回。

```
{
  "error_code" : "VOD.10053",
  "error_msg" : "The request parameter is illegal, illegal field: {xx}."
}
```

状态码

状态码	描述
204	处理成功返回。
400	处理失败返回。

错误码

请参见[错误码](#)。

12.3 查询水印列表

功能介绍

查询水印模板

URI

GET /v1.0/{project_id}/template/watermark

表 12-10 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

表 12-11 Query 参数

参数	是否必选	参数类型	描述
id	否	Array of strings	水印模板配置id，一次最多10个。
page	否	Integer	分页编号。 默认为0。指定id时该参数无效。

参数	是否必选	参数类型	描述
size	否	Integer	每页记录数。 默认为10，范围[1,100]。指定id时该参数无效。

请求参数

表 12-12 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

响应参数

状态码：200

表 12-13 响应 Body 参数

参数	参数类型	描述
templates	Array of WatermarkTemplate objects	水印模板信息。
total	Integer	总数。

表 12-14 WatermarkTemplate

参数	参数类型	描述
name	String	水印模板名称。
id	String	水印模板配置id。

参数	参数类型	描述
status	Long	启用状态。 取值为： <ul style="list-style-type: none"> 0：停用 1：启用
dx	String	水印图片相对输出视频的水平偏移量。 默认值是0。
dy	String	水印图片相对输出视频的垂直偏移量。 默认值是0。
position	String	水印的位置。
width	String	水印图片宽。
height	String	水印图片高。
create_time	String	创建时间。
image_url	String	水印图片下载url。
type	String	水印图片格式类型。
watermark_type	String	水印类型，当前只支持IMAGE（图片水印）。
image_process	String	watermark_type设置为IMAGE时有效。 目前包括： <ul style="list-style-type: none"> ORIGINAL：只做简单缩放，不做其他处理 TRANSPARENT：图片底色透明 GRAYED：彩色图片变灰
timeline_start	String	水印开始时间，与"timeline_duration"配合使用。 取值范围:[0, END)。"END"表示视频结束时间。 单位:秒。
timeline_duration	String	水印持续时间，与"timeline_start"配合使用。 取值范围:(0,END-开始时间]。"END"表示视频结束时间。 单位:秒。 默认:END。

状态码：400

表 12-15 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

GET https://{endpoint}/v1.0/{project_id}/template/watermark?id={id}

响应示例

状态码：200

处理成功返回。

```
{
  "templates": [ {
    "name": "test",
    "id": "2305739f855413a84af9e6ad6ebb21be",
    "status": 0,
    "dx": "0.05",
    "dy": "0.05",
    "position": "TOPRIGHT",
    "width": "0.12",
    "height": null,
    "create_time": "20210204092325",
    "image_url": "https://103-cn-north-4.cdn-vod.huaweicloud.com/05ab5cef408026f22f62c018de60cf2e/watermark/2305739f855413a84af9e6ad6ebb21be.png",
    "type": "PNG",
    "watermark_type": "IMAGE",
    "image_process": "TRANSPARENT",
    "timeline_start": null,
    "timeline_duration": null
  } ],
  "total": 1
}
```

状态码：400

处理失败返回。

```
{
  "error_code": "VOD.10053",
  "error_msg": "The request parameter is illegal, illegal field: {xx}."
}
```

状态码

状态码	描述
200	处理成功返回。
400	处理失败返回。

错误码

请参见[错误码](#)。

12.4 删除水印模板

功能介绍

删除水印模板

URI

DELETE /v1.0/{project_id}/template/watermark

表 12-16 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

表 12-17 Query 参数

参数	是否必选	参数类型	描述
id	是	String	水印模板配置id

请求参数

表 12-18 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

响应参数

状态码：400

表 12-19 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

删除水印模板

```
DELETE https://{endpoint}/v1.0/{project_id}/template/watermark?id={id}
```

响应示例

状态码：400

处理失败返回

```
{  
  "error_code": "VOD.10053",  
  "error_msg": "The request parameter is illegal, illegal field: {xx}."  
}
```

状态码

状态码	描述
204	处理成功返回
400	处理失败返回

错误码

请参见[错误码](#)。

12.5 确认水印图片上传

功能介绍

确认水印图片上传状态。

URI

POST /v1.0/{project_id}/watermark/status/uploaded

表 12-20 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

请求参数

表 12-21 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

表 12-22 请求 Body 参数

参数	是否必选	参数类型	描述
id	是	String	水印配置模板id。
status	是	String	水印上传状态，传值支持"SUCCEED"和"FAILED"。

响应参数

状态码：200

表 12-23 响应 Body 参数

参数	参数类型	描述
id	String	水印配置模板id
image_url	String	水印图片的下载url

状态码：400

表 12-24 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

确认水印图片上传状态

POST https://{endpoint}/v1.0/{project_id}/watermark/status/uploaded

```
Content-Type: application/json
{
  "id": "2305739f855413a84af9e6ad6e2b21be",
  "status": "SUCCEED"
}
```

响应示例

状态码：200

处理成功返回。

```
{
  "id": "2305739f855413a84af9e6ad6e2b21be",
  "image_url": "https://vod-bucket-26-cn-north-4.obs.cn-north-4.myhuaweicloud.com:443/05ab5cef408026f22f62c018de60cf2e/a52ba84366abebb4c4614e1b16973549/watermark.png?AWSAccessKeyId=CBN2j*****ORCSN&Expires=1625479312&Signature=kZYh0hEos2V*****AHGyXA%3D"
}
```

状态码：400

处理失败返回。

```
{
  "error_code": "VOD.10053",
  "error_msg": "The request parameter is illegal, illegal field: {xx}."
}
```

状态码

状态码	描述
200	处理成功返回。
400	处理失败返回。

错误码

请参见[错误码](#)。

13 字幕管理

13.1 多字幕封装

功能介绍

多字幕封装，支持HLS VTT格式和HLS SRT格式。

URI

PUT /v1/{project_id}/asset/subtitles

表 13-1 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

请求参数

表 13-2 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。

参数	是否必选	参数类型	描述
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

表 13-3 请求 Body 参数

参数	是否必选	参数类型	描述
asset_id	是	String	VOD分配的媒资ID，仅支持查询，不支持修改。
default_language	否	String	字幕默认语言（字幕必须存在）
repackage_mode	否	String	外挂模式，不传默认取值为0 取值如下： -0：表示添加的字幕会外挂上历史产物 -1：表示添加的字幕不会外挂上历史产物
delete_mode	否	String	删除模式，不传默认取值为0 取值如下： -0：表示删除字幕会清除历史产物携带的字幕信息 -1：表示删除字幕不清除历史产物携带的字幕信息
add_subtitles	否	Array of AddSubtitle objects	需新增或修改的字幕
delete_subtitles	否	Array of DeleteSubtitle objects	需删除的字幕，language不能与add_subtitles重复

表 13-4 AddSubtitle

参数	是否必选	参数类型	描述
type	是	String	字幕类型，字幕封装当前支持VTT和SRT
language	是	String	字幕语言
obs_info	是	ObsInfo object	字幕OBS桶信息

表 13-5 ObsInfo

参数	是否必选	参数类型	描述
bucket	是	String	OBS的bucket名称
object	是	String	OBS对象路径，需具体到对象

表 13-6 DeleteSubtitle

参数	是否必选	参数类型	描述
type	是	String	字幕类型，字幕封装当前支持VTT和SRT
language	是	String	字幕语言

响应参数

状态码：202

表 13-7 响应 Body 参数

参数	参数类型	描述
asset_id	String	VOD分配的媒资ID，仅支持查询，不支持修改。

状态码：400

表 13-8 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

多字幕封装

```
PUT https://{endpoint}/v1/{project_id}/asset/subtitles
```

```
Content-Type: application/json
```

```
{
  "asset_id": "f488337c31c8e4622f1590735b134c65",
  "default_language": "cn",
  "type": "custom_template_group",
  "is_default": true,
```

```

"add_subtitles": [
  {
    "obs_info": {
      "bucket": "test",
      "object": "subtitle_test/happyNewYearCn.vtt"
    },
    "type": "VTT",
    "language": "cn"
  }
],
"delete_subtitles": [
  {
    "type": "VTT",
    "language": "cn"
  }
]
}

```

响应示例

状态码：202

处理成功返回。

```

{
  "asset_id": "a3ef6526e787acff2ca81c7da840f11f"
}

```

状态码：400

处理失败返回。

```

{
  "error_code": "VOD.10053",
  "error_msg": "The request parameter is illegal, illegal field: {xx}."
}

```

状态码

状态码	描述
202	处理成功返回。
400	处理失败返回。

错误码

请参见[错误码](#)。

14 转码模板管理

14.1 创建自定义转码模板

功能介绍

创建自定义转码模板。

URI

POST /v2/{project_id}/asset/template/transcodings

表 14-1 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

请求参数

表 14-2 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。

参数	是否必选	参数类型	描述
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

表 14-3 请求 Body 参数

参数	是否必选	参数类型	描述
name	是	String	模板组名称
is_default	否	Boolean	是否设置成默认转码模板，默认false不设置为默认
is_auto_encrypt	否	Boolean	配置是否加密输出音视频，默认false不加密。 当前只支持加密输出HLS格式的音视频流，不会对原始上传的音视频进行加密处理。非HLS转码输出的音视频格式，此字段不可配置。开启此配置前，请参考《VOD用户指南》的“HLS加密设置”章节获取密钥URL。
quality_info_list	是	Array of QualityInfoList objects	画质配置信息列表
common	是	CommonInfo object	高清低码开关，编码格式多路共同参数
watermark_template_ids	否	Array of strings	绑定的水印模板组ID数组
description	否	String	模板介绍

表 14-4 QualityInfoList

参数	是否必选	参数类型	描述
video	否	VideoInfo object	模板视频信息。video和audio至少配置一个参数。
audio	否	AudioInfo object	模板音频信息

表 14-5 VideoInfo

参数	是否必选	参数类型	描述
quality	是	String	画质 <ul style="list-style-type: none"> • 4K默认分辨率3840*2160, 码率8000kbit/s • 2K默认分辨率2560*1440, 码率7000kbit/s • FULL_HD默认分辨率1920*1080, 码率3000kbit/s • HD默认分辨率1280*720, 码率1000kbit/s • SD默认分辨率854*480, 码率600kbit/s • FLUENT默认分辨率480*270, 码率300kbit/s
width	否	Integer	视频宽度 H264编码: 0或[32,4096]之间2的整数倍 H265编码: 0或[160,4096]之间2的整数倍
height	否	Integer	视频高度 H264编码: 0或[32,2880]之间2的整数倍 H265编码: 0或[160,2880]之间2的整数倍
bitrate	是	Integer	平均码率, 单位: kbit/s。 参考取值范围: 0或[40,30000]之间的整数。
frame_rate	是	Integer	帧率, 单位是帧每秒。 取值范围: [0-75]之间的整数, 小于5帧或大于60帧代表自适应。

表 14-6 AudioInfo

参数	是否必选	参数类型	描述
sample_rate	是	String	音频采样率（有效值范围） <ul style="list-style-type: none"> • AUDIO_SAMPLE_AUTO (default), • AUDIO_SAMPLE_22050: 22050Hz • AUDIO_SAMPLE_32000: 32000Hz • AUDIO_SAMPLE_44100: 44100Hz • AUDIO_SAMPLE_48000: 48000Hz • AUDIO_SAMPLE_96000: 96000Hz
bitrate	否	Integer	音频码率（单位：Kbps）
channels	是	String	声道数（有效值范围） <ul style="list-style-type: none"> • AUDIO_CHANNELS_1:单声道 • AUDIO_CHANNELS_2: 双声道 (default) • AUDIO_CHANNELS_5_1: 5.1声道

表 14-7 CommonInfo

参数	是否必选	参数类型	描述
pvc	否	Boolean	高清低码开关
video_codec	否	String	视频编码格式 <ul style="list-style-type: none"> • H264 • H265
audio_codec	否	String	音频编码格式 <ul style="list-style-type: none"> • AAC: AAC格式 (default) • HEAAC1: HEAAC1格式 • HEAAC2: HEAAC2格式 • MP3: MP3格式
is_black_cut	否	Boolean	黑边剪裁类型，默认false不开启黑边剪裁

参数	是否必选	参数类型	描述
format	是	String	转码格式 <ul style="list-style-type: none"> • MP4 • HLS • DASH • DASH_HLS • MP3 • ADTS • UNKNOWN
hls_interval	否	Integer	分片时长，范围2-10（默认为5秒）
upsample	否	Boolean	上采样开关，开启后可以提升视频的分辨率，增加采样点的数量，默认值:false，表示不开启上采样。
adaptation	否	String	转码后的片源分辨率自适应方式。 取值如下： <ul style="list-style-type: none"> • SHORT：短边自适应。转码模板的宽和高都设置，用其中的短边与片源短边对比，然后片源长边按此比例进行缩放。 • LONG：长边自适应。转码模板的宽和高都设置，用其中的长边与片源长边对比，然后片源短边按此比例进行缩放。 • NONE：按设置宽高适应，为默认值。按照转码模板设置的宽高输出。 说明 <ul style="list-style-type: none"> • V1接口（已废弃）不支持设置该参数，默认按SHORT处理。V2接口默认按NONE处理，V1和V2接口默认值有差异，在调用时需要注意该参数值的设置。 • 控制台是通过V2接口创建的转码模板，实际与V1接口创建转码模板的默认值保持一致，为SHORT。
preset	否	Integer	编码质量等级，取值[0,2] 0表示默认方式，1表示转码效率优先，2表示转码质量优先。

参数	是否必选	参数类型	描述
max_iframes_interval	否	Integer	I帧最大间隔，取值范围：[2, 10]。默认值：5，单位秒。
hls_audio_separate	否	Boolean	转码后音频是否独立存储。
hls_segment_type	否	String	HLS分片的封装格式，目前支持TS和FMP4，默认TS格式

响应参数

状态码：201

表 14-8 响应 Body 参数

参数	参数类型	描述
group_id	String	模板组ID

状态码：400

表 14-9 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

创建转码模板

POST https://{endpoint}/v2/{project_id}/asset/template/transcodings

Content-Type: application/json

```
{
  "name": "trans_template_test",
  "is_default": true,
  "is_auto_encrypt": false,
  "quality_info_list": [
    {
      "video": {
        "width": 1280,
        "height": 720,
        "bitrate": 1000,
        "quality": "HD",
        "frame_rate": 0
      },
      "audio": {
        "sample_rate": "AUDIO_SAMPLE_AUTO",
```

```
"channels": "AUDIO_CHANNELS_1",
"bitrate": 0
}
},
],
"watermark_template_ids": [],
"common": {
  "pvc": false,
  "video_codec": "H264",
  "audio_codec": "AAC",
  "format": "HLS",
  "hls_interval": 5
}
}
```

响应示例

状态码：201

处理成功返回。

```
{
  "group_id" : "f9b045e0811c482f9de0d436a5927bb6"
}
```

状态码：400

处理失败返回。

```
{
  "error_code" : "VOD.10053",
  "error_msg" : "The request parameter is illegal, illegal field: {xx}."
}
```

状态码

状态码	描述
201	处理成功返回。
400	处理失败返回。

错误码

请参见[错误码](#)。

14.2 查询转码模板列表

功能介绍

查询转码模板列表。

URI

GET /v2/{project_id}/asset/template/transcodings

表 14-10 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

表 14-11 Query 参数

参数	是否必选	参数类型	描述
group_id	否	String	模板id
is_default	否	Boolean	是否默认模板
offset	否	Integer	偏移量。默认为0。指定group_id时该参数无效。
limit	否	Integer	每页记录数。默认为10，范围[1,100]。指定group_id时该参数无效。
query_string	否	String	按照模板名和描述模糊查询。指定group_id时该参数无效。

请求参数

表 14-12 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

响应参数

状态码：200

表 14-13 响应 Body 参数

参数	参数类型	描述
template_group_list	Array of TransTemplateResp objects	模板组信息
total	Integer	总记录条数

表 14-14 TransTemplateResp

参数	参数类型	描述
group_id	String	模板组id
name	String	模板组名称
is_default	Boolean	是否设置成默认转码模板
type	String	模板组类型
is_auto_encrypt	Boolean	配置是否加密输出音视频，默认false不加密。 当前只支持加密输出HLS格式的音视频流，不会对原始上传的音视频进行加密处理。非HLS转码输出的音视频格式，此字段不可配置。开启此配置前，请参考《VOD用户指南》的“HLS加密设置”章节获取密钥URL。
quality_info_list	Array of QualityInfoList objects	画质配置信息列表
watermark_template_ids	Array of strings	绑定的水印模板组ID数组
description	String	模板介绍
common	CommonInfo object	高清低码开关，高清低码Version，编码格式多路共同参数

表 14-15 QualityInfoList

参数	参数类型	描述
video	VideoInfo object	模板视频信息。video和audio至少配置一个参数。
audio	AudioInfo object	模板音频信息

表 14-16 VideoInfo

参数	参数类型	描述
quality	String	<p>画质</p> <ul style="list-style-type: none"> ● 4K默认分辨率3840*2160, 码率8000kbit/s ● 2K默认分辨率2560*1440, 码率7000kbit/s ● FULL_HD默认分辨率1920*1080, 码率3000kbit/s ● HD默认分辨率1280*720, 码率1000kbit/s ● SD默认分辨率854*480, 码率600kbit/s ● FLUENT默认分辨率480*270, 码率300kbit/s
width	Integer	<p>视频宽度</p> <p>H264编码: 0或[32,4096]之间2的整数倍</p> <p>H265编码: 0或[160,4096]之间2的整数倍</p>
height	Integer	<p>视频高度</p> <p>H264编码: 0或[32,2880]之间2的整数倍</p> <p>H265编码: 0或[160,2880]之间2的整数倍</p>
bitrate	Integer	<p>平均码率, 单位: kbit/s。</p> <p>参考取值范围: 0或[40,30000]之间的整数。</p>
frame_rate	Integer	<p>帧率, 单位是帧每秒。</p> <p>取值范围: [0-75]之间的整数, 小于5帧或大于60帧代表自适应。</p>

表 14-17 AudioInfo

参数	参数类型	描述
sample_rate	String	音频采样率（有效值范围） <ul style="list-style-type: none"> AUDIO_SAMPLE_AUTO (default), AUDIO_SAMPLE_22050: 22050Hz AUDIO_SAMPLE_32000: 32000Hz AUDIO_SAMPLE_44100: 44100Hz AUDIO_SAMPLE_48000: 48000Hz AUDIO_SAMPLE_96000: 96000Hz
bitrate	Integer	音频码率（单位：Kbps）
channels	String	声道数（有效值范围） <ul style="list-style-type: none"> AUDIO_CHANNELS_1:单声道 AUDIO_CHANNELS_2: 双声道 (default) AUDIO_CHANNELS_5_1: 5.1声道

表 14-18 CommonInfo

参数	参数类型	描述
pvc	Boolean	高清低码开关
video_codec	String	视频编码格式 <ul style="list-style-type: none"> H264 H265
audio_codec	String	音频编码格式 <ul style="list-style-type: none"> AAC: AAC格式 (default) HEAAC1: HEAAC1格式 HEAAC2: HEAAC2格式 MP3: MP3格式
is_black_cut	Boolean	黑边剪裁类型，默认false不开启黑边剪裁

参数	参数类型	描述
format	String	转码格式 <ul style="list-style-type: none"> • MP4 • HLS • DASH • DASH_HLS • MP3 • ADTS • UNKNOW
hls_interval	Integer	分片时长, 范围2-10 (默认为5秒)
upsample	Boolean	上采样开关, 开启后可以提升视频的分辨率, 增加采样点的数量, 默认值:false, 表示不开启上采样。
adaptation	String	转码后的片源分辨率自适应方式。 取值如下: <ul style="list-style-type: none"> • SHORT: 短边自适应。转码模板的宽和高都设置, 用其中的短边与片源短边对比, 然后片源长边按此比例进行缩放。 • LONG: 长边自适应。转码模板的宽和高都设置, 用其中的长边与片源长边对比, 然后片源短边按此比例进行缩放。 • NONE: 按设置宽高适应, 为默认值。按照转码模板设置的宽高输出。 说明 <ul style="list-style-type: none"> • V1接口 (已废弃) 不支持设置该参数, 默认按SHORT处理。V2接口默认按NONE处理, V1和V2接口默认值有差异, 在调用时需要注意该参数值的设置。 • 控制台是通过V2接口创建的转码模板, 实际与V1接口创建转码模板的默认值保持一致, 为SHORT。
preset	Integer	编码质量等级, 取值[0,2] 0表示默认方式, 1表示转码效率优先, 2表示转码质量优先。
max_iframes_interval	Integer	I帧最大间隔, 取值范围: [2, 10]。默认值: 5, 单位秒。
hls_audio_separate	Boolean	转码后音频是否独立存储。
hls_segment_type	String	HLS分片的封装格式, 目前支持TS和FMP4, 默认TS格式

状态码：403

表 14-19 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

查询转码模板

GET https://{endpoint}/v2/{project_id}/asset/template/transcodings?group_id={group_id}

响应示例

状态码：200

处理成功返回。

```
{
  "template_group_list": [ {
    "group_id": "9751249d25f14587b212544d6fd8dcf8",
    "name": "non_transcoding_template_group",
    "is_default": false,
    "type": "non_transcoding_template_group",
    "is_auto_encrypt": false,
    "quality_info_list": [ {
      "video": {
        "quality": "UNKNOWN",
        "width": 0,
        "height": 0,
        "bitrate": 0,
        "frame_rate": 0
      },
      "audio": null
    } ],
    "watermark_template_ids": null,
    "description": null,
    "common": {
      "pvc": false,
      "is_black_cut": false,
      "format": "UNKNOWN",
      "upsample": false,
      "adaptation": "NONE",
      "video_codec": null,
      "audio_codec": "AAC",
      "hls_interval": 0
    }
  } ],
  "total": 0
}
```

状态码：403

处理失败返回。

```
{
  "error_code": "VOD.10053",
```



```
"error_msg" : "The request parameter is illegal, illegal field: {xx}."
}
```

状态码

状态码	描述
200	处理成功返回。
403	处理失败返回。

错误码

请参见[错误码](#)。

14.3 修改转码模板

功能介绍

修改转码模板。

URI

PUT /v2/{project_id}/asset/template/transcodings

表 14-20 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

请求参数

表 14-21 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。

参数	是否必选	参数类型	描述
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

表 14-22 请求 Body 参数

参数	是否必选	参数类型	描述
group_id	是	String	模板组ID
name	否	String	模板组名称
is_default	否	Boolean	是否是默认转码模板，默认false不设置为默认
is_auto_encrypt	否	Boolean	配置是否加密输出音视频，默认false不加密。 当前只支持加密输出HLS格式的音视频流，不会对原始上传的音视频进行加密处理。非HLS转码输出的音视频格式，此字段不可配置。开启此配置前，请参考《VOD用户指南》的“HLS加密设置”章节获取密钥URL。
quality_info_list	否	Array of QualityInfoList objects	画质配置信息列表，如果不携带或者携带为空，则不更新模板中该部分数据。
watermark_template_ids	否	Array of strings	绑定的水印模板组ID数组
description	否	String	模板介绍
common	否	CommonInfo object	高清低码开关，编码格式多路共同参数

表 14-23 QualityInfoList

参数	是否必选	参数类型	描述
video	否	VideoInfo object	模板视频信息。video和audio至少配置一个参数。
audio	否	AudioInfo object	模板音频信息

表 14-24 VideoInfo

参数	是否必选	参数类型	描述
quality	是	String	画质 <ul style="list-style-type: none"> • 4K默认分辨率3840*2160, 码率8000kbit/s • 2K默认分辨率2560*1440, 码率7000kbit/s • FULL_HD默认分辨率1920*1080, 码率3000kbit/s • HD默认分辨率1280*720, 码率1000kbit/s • SD默认分辨率854*480, 码率600kbit/s • FLUENT默认分辨率480*270, 码率300kbit/s
width	否	Integer	视频宽度 H264编码: 0或[32,4096]之间2的整数倍 H265编码: 0或[160,4096]之间2的整数倍
height	否	Integer	视频高度 H264编码: 0或[32,2880]之间2的整数倍 H265编码: 0或[160,2880]之间2的整数倍
bitrate	是	Integer	平均码率, 单位: kbit/s。 参考取值范围: 0或[40,30000]之间的整数。
frame_rate	是	Integer	帧率, 单位是帧每秒。 取值范围: [0-75]之间的整数, 小于5帧或大于60帧代表自适应。

表 14-25 AudioInfo

参数	是否必选	参数类型	描述
sample_rate	是	String	音频采样率（有效值范围） <ul style="list-style-type: none"> • AUDIO_SAMPLE_AUTO (default), • AUDIO_SAMPLE_22050: 22050Hz • AUDIO_SAMPLE_32000: 32000Hz • AUDIO_SAMPLE_44100: 44100Hz • AUDIO_SAMPLE_48000: 48000Hz • AUDIO_SAMPLE_96000: 96000Hz
bitrate	否	Integer	音频码率（单位：Kbps）
channels	是	String	声道数（有效值范围） <ul style="list-style-type: none"> • AUDIO_CHANNELS_1:单声道 • AUDIO_CHANNELS_2: 双声道 (default) • AUDIO_CHANNELS_5_1: 5.1声道

表 14-26 CommonInfo

参数	是否必选	参数类型	描述
pvc	否	Boolean	高清低码开关
video_codec	否	String	视频编码格式 <ul style="list-style-type: none"> • H264 • H265
audio_codec	否	String	音频编码格式 <ul style="list-style-type: none"> • AAC: AAC格式 (default) • HEAAC1: HEAAC1格式 • HEAAC2: HEAAC2格式 • MP3: MP3格式
is_black_cut	否	Boolean	黑边剪裁类型，默认false不开启黑边剪裁

参数	是否必选	参数类型	描述
format	是	String	转码格式 <ul style="list-style-type: none"> • MP4 • HLS • DASH • DASH_HLS • MP3 • ADTS • UNKNOW
hls_interval	否	Integer	分片时长，范围2-10（默认为5秒）
upsample	否	Boolean	上采样开关，开启后可以提升视频的分辨率，增加采样点的数量，默认值:false，表示不开启上采样。
adaptation	否	String	转码后的片源分辨率自适应方式。 取值如下： <ul style="list-style-type: none"> • SHORT：短边自适应。转码模板的宽和高都设置，用其中的短边与片源短边对比，然后片源长边按此比例进行缩放。 • LONG：长边自适应。转码模板的宽和高都设置，用其中的长边与片源长边对比，然后片源短边按此比例进行缩放。 • NONE：按设置宽高适应，为默认值。按照转码模板设置的宽高输出。 说明 <ul style="list-style-type: none"> • V1接口（已废弃）不支持设置该参数，默认按SHORT处理。V2接口默认按NONE处理，V1和V2接口默认值有差异，在调用时需要注意该参数值的设置。 • 控制台是通过V2接口创建的转码模板，实际与V1接口创建转码模板的默认值保持一致，为SHORT。
preset	否	Integer	编码质量等级，取值[0,2] 0表示默认方式，1表示转码效率优先，2表示转码质量优先。

参数	是否必选	参数类型	描述
max_iframes_interval	否	Integer	I帧最大间隔，取值范围：[2, 10]。默认值：5，单位秒。
hls_audio_separate	否	Boolean	转码后音频是否独立存储。
hls_segment_type	否	String	HLS分片的封装格式，目前支持TS和FMP4，默认TS格式

响应参数

状态码：400

表 14-27 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

修改转码模板

PUT https://{endpoint}/v2/{project_id}/asset/template/transcodings

Content-Type: application/json

```
{
  "group_id": "f9b045e0811c482f9de0d436a5927bb6",
  "name": "trans_template_test",
  "is_default": true,
  "quality_info_list": [
    {
      "video": {
        "width": 1280,
        "height": 720,
        "bitrate": 1000,
        "quality": "HD",
        "frame_rate": 0
      },
      "audio": {
        "sample_rate": "AUDIO_SAMPLE_AUTO",
        "channels": "AUDIO_CHANNELS_1",
        "bitrate": 0
      }
    }
  ],
  "watermark_template_ids": [],
  "common": {
    "pvc": false,
    "video_codec": "H264",
    "audio_codec": "AAC",
    "format": "HLS",
    "hls_interval": 5
  }
}
```

响应示例

状态码：400

处理失败返回。

```
{
  "error_code": "VOD.10053",
  "error_msg": "The request parameter is illegal, illegal field: {xx}."
}
```

状态码

状态码	描述
200	处理成功返回。
400	处理失败返回。

错误码

请参见[错误码](#)。

14.4 删除自定义模板

功能介绍

删除自定义模板。

URI

DELETE /v2/{project_id}/asset/template/transcodings

表 14-28 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

表 14-29 Query 参数

参数	是否必选	参数类型	描述
group_id	是	String	模板id

请求参数

表 14-30 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

响应参数

状态码：400

表 14-31 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

删除转码模板

```
DELETE https://{endpoint}/v2/{project_id}/asset/template/transcodings?group_id={group_id}
```

响应示例

状态码：400

处理失败返回。

```
{
  "error_code" : "VOD.10053",
  "error_msg" : "The request parameter is illegal, illegal field: {xx}."
}
```

状态码

状态码	描述
204	处理成功返回204 No Content。

状态码	描述
400	处理失败返回。

错误码

请参见[错误码](#)。

15 转码模板集合管理

15.1 创建转码模板组集合

功能介绍

创建转码模板组集合。

接口约束

- 不同模板的转码产物类型完全一样时，不支持添加至同一个转码模板组中。
- 相同编码格式的模板，在转码模板组的产物类型，不允许同时出现HLS/DASH（二选一）和HLS_DASH。不同编码格式的模板无影响。

其中codec为视频编码格式，取值为H.264或H.265，详见[查询媒资信息](#)。

URI

POST /v1.0/{project_id}/asset/template-collection/transcodings

表 15-1 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

请求参数

表 15-2 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

表 15-3 请求 Body 参数

参数	是否必选	参数类型	描述
name	是	String	模板组集合名称
description	否	String	模板组集合描述
template_group_list	是	Array of strings	模板组列表，模板ID

响应参数

状态码：201

表 15-4 响应 Body 参数

参数	参数类型	描述
group_collection_id	String	模板组集合ID

状态码：400

表 15-5 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

创建转码模板集合

```
POST https://{endpoint}/v1.0/{project_id}/asset/template-collection/transcodings

Content-Type: application/json
{
  "name": "test",
  "template_group_list": [
    "780640dd1d584a6999b104568c358b78",
    "6a16d8d0161c42caa42b9c148d032871"
  ]
}
```

响应示例

状态码：201

处理成功返回。

```
{
  "group_collection_id": "f9b045e0811c482f9de0d436a5927bb6"
}
```

状态码：400

处理失败返回。

```
{
  "error_code": "VOD.10053",
  "error_msg": "The request parameter is illegal, illegal field: {xx}."
}
```

状态码

状态码	描述
201	处理成功返回。
400	处理失败返回。

错误码

请参见[错误码](#)。

15.2 修改转码模板组集合

功能介绍

修改转码模板组集合。

URI

PUT /v1.0/{project_id}/asset/template-collection/transcodings

表 15-6 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

请求参数

表 15-7 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

表 15-8 请求 Body 参数

参数	是否必选	参数类型	描述
name	否	String	模板组集合名称
collection_id	是	String	模板组集合ID
description	否	String	模板组集合介绍
template_group_list	否	Array of strings	模板组列表

响应参数

状态码：400

表 15-9 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

修改转码模板集合

```
POST https://{endpoint}/v1.0/{project_id}/asset/template-collection/transcodings
```

```
Content-Type: application/json
```

```
{
  "collection_id": "f9b045e0811c482f9de0d436a5927bb6",
  "name": "test",
  "template_group_list": [
    "780640dd1d584a6999b104568c358b78",
    "6a16d8d0161c42caa42b9c148d032871"
  ]
}
```

响应示例

状态码：400

处理失败返回。

```
{
  "error_code": "VOD.10053",
  "error_msg": "The request parameter is illegal, illegal field: {xx}."
}
```

状态码

状态码	描述
200	处理成功返回。
400	处理失败返回。

错误码

请参见[错误码](#)。

15.3 查询自定义模板组集合

功能介绍

查询转码模板组集合。

URI

GET /v1.0/{project_id}/asset/template-collection/transcodings

表 15-10 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

表 15-11 Query 参数

参数	是否必选	参数类型	描述
group_collection_id	否	String	模板组集合id
offset	否	Integer	偏移量。默认为0。指定group_collection_id时该参数无效。
limit	否	Integer	每页记录数。默认为10，范围[1,100]。指定group_collection_id时该参数无效。

请求参数

表 15-12 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

响应参数

状态码：200

表 15-13 响应 Body 参数

参数	参数类型	描述
template_group_collection_list	Array of TemplateGroupCollection objects	模板组集合信息
total	Integer	总记录条数

表 15-14 TemplateGroupCollection

参数	参数类型	描述
group_collection_id	String	模板组集合id
name	String	模板组集合名称
description	String	模板介绍
template_group_list	Array of TemplateGroup objects	转码组列表

表 15-15 TemplateGroup

参数	参数类型	描述
group_id	String	模板组id
name	String	模板组名称
status	String	是否默认
type	String	模板组类型
auto_encrypt	Integer	是否自动加密。 取值如下： <ul style="list-style-type: none"> 0：表示不加密。 1：表示需要加密。 默认值：0。 加密与转码必须要一起进行，当需要加密时，转码参数不能为空，且转码输出格式必须要为HLS。
quality_info_list	Array of QualityInfo objects	画质配置信息列表

参数	参数类型	描述
watermark_template_ids	Array of strings	绑定的水印模板组ID数组
description	String	模板介绍
common	Common object	高清低码开关，高清低码版本，编码格式多路共同参数

表 15-16 QualityInfo

参数	参数类型	描述
video	VideoTemplateInfo object	模板视频信息
audio	AudioTemplateInfo object	模板音频信息
format	String	格式。

表 15-17 VideoTemplateInfo

参数	参数类型	描述
quality	String	画质。
width	Integer	视频宽度。
height	Integer	视频高度。
bitrate	Integer	码率。
frame_rate	Integer	帧率（默认为1，1代表自适应，单位是帧每秒）。

表 15-18 AudioTemplateInfo

参数	参数类型	描述
sample_rate	Integer	音频采样率（有效值范围） <ul style="list-style-type: none"> 1: AUDIO_SAMPLE_AUTO 2: AUDIO_SAMPLE_22050 3: AUDIO_SAMPLE_32000 4: AUDIO_SAMPLE_44100 5: AUDIO_SAMPLE_48000 6: AUDIO_SAMPLE_96000 默认值为1。
bitrate	Integer	音频码率（单位：Kbps）。
channels	Integer	声道数（有效值范围） <ul style="list-style-type: none"> 1: AUDIO_CHANNELS_1 2: AUDIO_CHANNELS_2

表 15-19 Common

参数	参数类型	描述
pvc	String	高清低码开关。
pvc_version	String	高清低码版本。
video_codec	String	视频编码格式。
audio_codec	String	音频编码格式（有效值范围） <ul style="list-style-type: none"> 1: AUDIO_CODECTYPE_AAC 2: AUDIO_CODECTYPE_HEAAC1 3: AUDIO_CODECTYPE_HEAAC2 4: AUDIO_CODECTYPE_MP3 默认值为1。
hls_interval	Integer	分片时长（默认为5秒）。

状态码：400

表 15-20 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

查询转码模板集合

GET https://{endpoint}/v1.0/{project_id}/asset/template-collection/transcodings

响应示例

状态码：200

处理成功返回。

```
{
  "template_group_collection_list" : [ {
    "group_collection_id" : "9751249d25f14587b212544d6fd8dcf8",
    "name" : "test",
    "template_group_list" : [ {
      "group_id" : "9751249d25f14587b212544d6fd8dcf8",
      "name" : "test112",
      "status" : "0",
      "type" : "custom_template_group",
      "auto_encrypt" : 0,
      "quality_info_list" : [ {
        "video" : {
          "quality" : "UNKNOW",
          "width" : 0,
          "height" : 0,
          "bitrate" : 0,
          "frame_rate" : 0
        },
        "audio" : null,
        "format" : "UNKNOW"
      } ],
      "watermark_template_ids" : null,
      "description" : null,
      "common" : {
        "pvc" : null,
        "pvc_version" : null,
        "video_codec" : null,
        "audio_codec" : "AAC",
        "hls_interval" : 0
      }
    } ]
  } ],
  "total" : 1
}
```

状态码：400

处理失败返回。

```
{
  "error_code" : "VOD.10053",
  "error_msg" : "The request parameter is illegal, illegal field: {xx}."
}
```

状态码

状态码	描述
200	处理成功返回。

状态码	描述
400	处理失败返回。

错误码

请参见[错误码](#)。

15.4 删除转码模板组集合

功能介绍

删除转码模板组集合

URI

DELETE /v1.0/{project_id}/asset/template-collection/transcodings

表 15-21 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

表 15-22 Query 参数

参数	是否必选	参数类型	描述
group_collection_id	是	String	模板组集合id

请求参数

表 15-23 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。

参数	是否必选	参数类型	描述
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

响应参数

状态码：400

表 15-24 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

删除转码模板集合

```
DELETE https://{endpoint}/v1.0/{project_id}/asset/template-collection/transcodings?group_collection_id={group_collection_id}
```

响应示例

状态码：400

处理失败返回。

```
{
  "error_code" : "VOD.10053",
  "error_msg" : "The request parameter is illegal, illegal field: {xx}."
}
```

状态码

状态码	描述
204	处理成功返回。
400	处理失败返回。

错误码

请参见[错误码](#)。

16 转码产物管理

16.1 删除转码产物

功能介绍

删除转码产物。

URI

DELETE /v1/{project_id}/asset/transcode-product

表 16-1 路径参数

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

请求参数

表 16-2 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

表 16-3 请求 Body 参数

参数	是否必选	参数类型	描述
asset_id	否	String	媒资ID。
delete_type	否	String	GROUP: 模板组级删除。 PRODUCT: 产物级删除
delete_infos	否	Array of ProductGroupInfo objects	删除的产物信息，最多传100个产物信息。

表 16-4 ProductGroupInfo

参数	是否必选	参数类型	描述
group_id	否	String	模板组ID
products	否	Array of ProductUrlInfo objects	产物信息

表 16-5 ProductUrlInfo

参数	是否必选	参数类型	描述
url	否	String	删除的产物URL

响应参数

状态码: 200

表 16-6 响应 Body 参数

参数	参数类型	描述
asset_id	String	VOD分配的媒资ID，仅支持查询，不支持修改。
status	String	SUCCESS: 成功 FAIL: 失败 PARTIAL_SUCCESS: 部分成功

参数	参数类型	描述
deleted_products	Array of ProductGroupInfo objects	删除成功的产物
failed_products	Array of ProductGroupDelFailInfo objects	删除失败的产物

表 16-7 ProductGroupInfo

参数	参数类型	描述
group_id	String	模板组ID
products	Array of ProductUrlInfo objects	产物信息

表 16-8 ProductUrlInfo

参数	参数类型	描述
url	String	删除的产物URL

表 16-9 ProductGroupDelFailInfo

参数	参数类型	描述
group_id	String	模板组ID
fail_reason	String	模板组删除失败的原因
products	Array of ProductDelFailInfo objects	删除失败的产物的信息

表 16-10 ProductDelFailInfo

参数	参数类型	描述
url	String	删除产物的URL
fail_reason	String	删除产物失败的原因

状态码：403

表 16-11 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

- DELETE https://{endpoint}/v1/{project_id}/asset/transcode-product

Content-Type: application/json

```
{
  "asset_id": "8c53b4038ea3d8694f1a8a7ac95f612f",
  "delete_type": "PRODUCT",
  "delete_infos": [
    {
      "group_id": "9171008cca7f47909c61f61a65d4c906",
      "products": [
        {
          "url": "https://103-cn-north-4.cdn-vod.huaweicloud.com/asset/8c53b4038ea3d8694f1a8a7ac95f612f/play_multi_video/9171008cca7f47909c61f61a65d4c906/38a92b74ead764ad21aa5355bad5193e.mp4"
        }
      ]
    }
  ]
}
```
- DELETE https://{endpoint}/v1/{project_id}/asset/transcode-product

Content-Type: application/json

```
{
  "asset_id": "6f306b4cc58c39b2a97ad797367b88ad",
  "delete_type": "GROUP",
  "delete_infos": [
    {
      "group_id": "02e6887dd6bf48299922eec52d192115"
    }
  ]
}
```

响应示例

状态码：200

处理成功返回。

```
{
  "status": "SUCCESS",
  "deleted_products": [ {
    "group_id": "9171008cca7f47909c61f61a65d4c906",
    "products": [ {
      "url": "https://103-cn-north-4.cdn-vod.huaweicloud.com/asset/8c53b4038ea3d8694f1a8a7ac95f612f/play_multi_video/9171008cca7f47909c61f61a65d4c906/da6613ac132eda54b2f4ab67adbf56bd.mp4"
    } ]
  } ],
  "failed_products": [ ]
}
```

状态码：403

处理失败返回。

```
{  
  "error_code" : "VOD.10053",  
  "error_msg" : "The request parameter is illegal, illegal field: {xx}."  
}
```

状态码

状态码	描述
200	处理成功返回。
403	处理失败返回。

错误码

请参见[错误码](#)。

17 媒资存储模式管理

17.1 修改媒资降冷粒度

功能介绍

修改媒资降冷粒度。

默认为整个媒资粒度降冷，也支持修改为原文件粒度降冷。

URI

PUT /v1/{project_id}/asset/storage-mode-type

表 17-1 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

请求参数

表 17-2 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。

参数	是否必选	参数类型	描述
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

表 17-3 请求 Body 参数

参数	是否必选	参数类型	描述
storage_mode_type	否	String	降冷模式。 取值如下： <ul style="list-style-type: none"> • WHOLE：整个媒资粒度降冷，即媒资原始文件、转码后的文件等整个媒资文件，均进行降冷处理。 • ORIGIN：原文件粒度降冷，即仅媒资原始视频文件降冷，其转码后的文件等，均不做降冷处理。

响应参数

状态码：400

表 17-4 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

修改媒资降冷粒度

```
PUT https://{endpoint}/v1/{project_id}/asset/storage-mode-type
Content-Type: application/json
{
  "storage_mode_type": "ORIGIN"
}
```

响应示例

状态码：400

处理失败返回。

```
{
  "error_code" : "VOD.10053",
```

```
"error_msg": "The request parameter is illegal, illegal field: {xx}."
}
```

状态码

状态码	描述
204	处理成功返回。
400	处理失败返回。

错误码

请参见[错误码](#)。

17.2 修改媒资文件在 OBS 的存储模式

功能介绍

修改媒资文件在OBS的存储模式。

媒资上传默认为标准存储模式，支持转为低频或者归档存储模式。

- 低频存储可恢复为标准存储或者降级为归档存储。
- 归档存储仅支持恢复为标准存储，不支持直接转为低频存储。
- 当媒资处于截图、转码等任务处理中状态时无法修改存储模式。

OBS存储计费注意事项，如下所示：

- 低频模式最低存储时间为30天，归档模式最低存储时间为90天。未满此时间提前取回或删除媒资文件，仍会按最低存储时间收费。
以低频存储模式为例进行说明，如果存储时间为28天（<30天），则按照最低存储时间30天计费；如果存储时间为31天（≥30天），则按实际天数计费。
- 媒资从低频或归档模式恢复成标准模式也会收取媒资取回的费用。

URI

PUT /v1/{project_id}/asset/storage-mode

表 17-5 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

请求参数

表 17-6 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

表 17-7 请求 Body 参数

参数	是否必选	参数类型	描述
asset_id	是	String	原媒资ID
storage_mode	是	String	存储模式。 取值如下： <ul style="list-style-type: none"> • STANDARD：标准存储。 • WARM：低频存储。 • COLD：归档存储。
restore_mode	否	String	归档恢复方式。 取值如下： <ul style="list-style-type: none"> • TEMP：临时 • FOREVER：永久
days	否	Integer	从归档存储临时恢复为标准存储的时间。 取值范围：1-30。单位：天。
restore_tier	否	String	归档恢复选项。 <ul style="list-style-type: none"> • EXPEDITED：快速恢复 • STANDARD：标准恢复 默认值：EXPEDITED。

响应参数

状态码：200

表 17-8 响应 Body 参数

参数	参数类型	描述
task_result_array	Array of TaskResult objects	修改媒资存储模式任务的下发结果。

表 17-9 TaskResult

参数	参数类型	描述
asset_id	String	媒资ID
status	String	查看修改媒资存储模式的任务是否下发成功。 <ul style="list-style-type: none"> • SUCCEEDED: 成功 • FAILED: 失败

状态码: 400

表 17-10 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

修改媒资文件在OBS的存储模式

```
PUT https://{endpoint}/v1/{project_id}/asset/storage-mode
```

```
Content-Type: application/json
```

```
{
  "asset_id": "2305739f855413a84af9e6ad6e2b21be",
  "storage_mode": "WARM"
}
```

响应示例

状态码: 200

处理成功返回。

```
{
  "task_result_array": [ {
    "asset_id": "2305739f855413a84af9e6ad6e2b21be",
    "status": "SUCCEEDED"
  } ]
}
```

状态码：400

处理失败返回。

```
{
  "error_code" : "VOD.10053",
  "error_msg" : "The request parameter is illegal, illegal field: {xx}."
}
```

状态码

状态码	描述
200	处理成功返回。
400	处理失败返回。

错误码

请参见[错误码](#)。

17.3 查询媒资降冷配置

功能介绍

查询媒资降冷配置。

URI

GET /v1/{project_id}/asset/storage-mode-type

表 17-11 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

请求参数

表 17-12 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。

参数	是否必选	参数类型	描述
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

响应参数

状态码：200

表 17-13 响应 Body 参数

参数	参数类型	描述
storage_mode_type	String	降冷模式。 取值如下： <ul style="list-style-type: none"> WHOLE：整个媒资粒度降冷，即媒资原始文件、转码后的文件等整个媒资文件，均进行降冷处理。 ORIGIN：原文件粒度降冷，即仅媒资原始视频文件降冷，其转码后的文件等，均不做降冷处理。

状态码：400

表 17-14 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

查询媒资降冷粒度

```
GET https://{endpoint}/v1/{project_id}/asset/storage-mode-type
```

响应示例

状态码：200

处理成功返回。

```
{  
  "storage_mode_type": "ORIGIN"  
}
```

状态码：400

处理失败返回。

```
{  
  "error_code": "VOD.10053",  
  "error_msg": "The request parameter is illegal, illegal field: {xx}."  
}
```

状态码

状态码	描述
200	处理成功返回。
400	处理失败返回。

错误码

请参见[错误码](#)。

17.4 查询取回数据信息

功能介绍

典型场景

用于查询点播低频和归档取回量统计数据。

接口功能

用于查询点播低频和归档取回量统计数据。

URI

GET /v1/{project_id}/asset/vod-retrieval

表 17-15 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

表 17-16 Query 参数

参数	是否必选	参数类型	描述
start_time	否	String	开始时间
end_time	否	String	结束时间
interval	否	Integer	<p>采样间隔，单位：秒，取值说明：</p> <p>时间跨度1天：1小时、4小时、8小时，分别对应3600秒、14400秒和28800秒。</p> <p>时间跨度2~7天：1小时、4小时、8小时、1天，分别对应3600秒、14400秒、28800秒和86400秒。</p> <p>时间跨度8~31天：4小时、8小时、1天，分别对应14400秒、28800秒和86400秒。</p> <p>如果不传，默认取对应时间跨度的最小间隔。</p>

请求参数

表 17-17 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	<p>用户Token，使用Token鉴权方式时必选。</p> <p>通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。</p>
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

响应参数

状态码：200

表 17-18 响应 Body 参数

参数	参数类型	描述
start_time	String	统计起始时间
interval	Integer	采样时间间隔
sample_data	Array of VodRetrievalData objects	

表 17-19 VodRetrievalData

参数	参数类型	描述
retrieval_warm	Double	低频存储的媒资恢复为标准存储后的文件大小
retrieval_cold	Double	归档存储的媒资按标准取回模式取回的文件大小
retrieval_cold_speed	Double	归档存储的媒资按快速取回模式取回的文件大小

状态码：400

表 17-20 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

GET https://{endpoint}/v1/{project_id}/asset/vod-retrieval

响应示例

状态码：200

处理成功返回。

```
{
  "start_time": "20240615000000",
  "interval": 3600,
  "sample_data": [ {
    "retrieval_warm": 1.88,
    "retrieval_cold": 2.4,
    "retrieval_cold_speed": 1.9
  }, {
```

```

"retrieval_warm" : 3.6,
"retrieval_cold" : 4.6,
"retrieval_cold_speed" : 3.56
}, {
"retrieval_warm" : 2.5,
"retrieval_cold" : 0,
"retrieval_cold_speed" : 0.08
}, {
"retrieval_warm" : 0,
"retrieval_cold" : 0,
"retrieval_cold_speed" : 0
}
}
    
```

状态码：400

处理失败返回。

```

{
"error_code" : "VOD.10053",
"error_msg" : "The request parameter is illegal, illegal field: {xx}."
}
    
```

状态码

状态码	描述
200	处理成功返回。
400	处理失败返回。

错误码

请参见[错误码](#)。

18 历史 API

18.1 媒资上传

18.1.1 获取分段上传授权

功能介绍

客户端请求创建媒资时，如果媒资文件超过20MB，需采用分段的方式向OBS上传，在每次与OBS交互前，客户端需通过此接口获取到授权方可与OBS交互。

该接口可以获取初始化多段上传任务、上传段、合并段、列举已上传段、取消段合并的带有临时授权的URL，用户需要根据OBS的接口文档配置相应请求的HTTP请求方法、请求头、请求体，然后请求对应的带有临时授权的URL。

视频分段上传方式和OBS的接口文档保持一致，包括HTTP请求方法、请求头、请求体等各种入参，此接口的作用是为用户生成鉴权信息（鉴权信息即sign_str），用来替换OBS接口中对应的URL，临时给用户开通点播服务的桶上传文件的权限。

调用获取授权接口时需要传入bucket、object_key、http_verb，其中bucket和object_key是由[创建媒资：上传方式](#)接口中返回的响应体中的target字段获得的bucket和object，http_verb需要根据指定的操作选择。

URI

GET /v1.0/{project_id}/asset/authority

表 18-1 路径参数

参数	是否必选	参数类型	描述
project_id	是	String	项目ID，获取方法请参考 获取项目ID

表 18-2 Query 参数

参数	是否必选	参数类型	描述
http_verb	是	String	分段上传时调用OBS接口的HTTP方法，具体操作需要的HTTP方法请参考OBS的接口文档。 <ul style="list-style-type: none"> 初始化上传任务：POST 上传段：PUT 合并段：POST 取消段：DELETE 列举已上传段：GET
bucket	是	String	桶名。 调用 创建媒资：上传方式 接口中返回的响应体中的target字段获得的bucket值。
object_key	是	String	对象名。 调用 创建媒资：上传方式 接口中返回的响应体中的target字段获得的object值。
content_type	否	String	文件类型对应的content-type，当为上传任务初始化场景该字段必填。 参数填写可参考 媒资分段上传（20M以上） 。 <ul style="list-style-type: none"> 视频文件：video/视频格式，如：“video/mp4” 音频文件：audio/音频格式，如：“audio/mp3” 图片文件：image/图片格式，如：“image/png” 字幕文件：application/octet-stream
content_md5	否	String	上传段时每段的MD5。
upload_id	否	String	每一个上传任务的id，是OBS初始化视频分段上传任务后返回的值。除上传任务初始化场景外，该字段必填。
part_number	否	Integer	上传段时每一段的id。 取值范围：[1,10000]。

请求参数

表 18-3 请求 Header 参数

参数	是否必选	参数类型	描述
X-Auth-Token	否	String	用户Token，使用Token鉴权方式时必选。 通过调用IAM服务获取用户Token接口获取（响应消息头中X-Subject-Token的值）。
Authorization	否	String	使用AK/SK方式认证时必选，携带的鉴权信息。
X-Sdk-Date	否	String	使用AK/SK方式认证时必选，请求的发生时间。

响应参数

状态码： 200

表 18-4 响应 Body 参数

参数	参数类型	描述
sign_str	String	授权签名字符串。 示例： AWSAccessKeyId={AccessKeyID}&Expires={ExpiresValue}&Signature={Signature}

状态码： 403

表 18-5 响应 Body 参数

参数	参数类型	描述
error_code	String	错误码。
error_msg	String	错误描述。

请求示例

- 初始化上传任务：
GET https://{endpoint}/v1.0/{project_id}/asset/authority?
http_verb=POST&content_type={type}&bucket={bucket}&object_key={objectKey}
- 上传段：


```
GET https://{endpoint}/v1.0/{project_id}/asset/authority?
http_verb=PUT&content_md5={md5}&part_number={num}&upload_id={id}&bucket={bucket}&object_key={objectKey}
```

- 合并段:

```
GET https://{endpoint}/v1.0/{project_id}/asset/authority?
http_verb=POST&upload_id={id}&bucket={bucket}&object_key={objectKey}
```

- 取消段:

```
GET https://{endpoint}/v1.0/{project_id}/asset/authority?
http_verb=DELETE&bucket={bucket}&object_key={objectKey}&upload_id={uploadId}
```

- 列举已上传的段:

```
GET https://{endpoint}/v1.0/{project_id}/asset/authority?
http_verb=GET&bucket={bucket}&object_key={objectKey}&upload_id={uploadId}
```

响应示例

状态码: 200

处理成功返回。

```
{
  "sign_str" :
  "AWSAccessKeyId=UBWV*****N4NOJ&Expires=1730739600&Signature=47rXht3Qrd*****GilxM40%3D"
}
```

状态码

状态码	描述
200	处理成功返回。
403	处理失败返回。

错误码

请参见[错误码](#)。

19 附录

19.1 状态码

调用视频点播API时，可能返回的状态码如表19-1所示。

表 19-1 状态码

返回值	说明
200 OK	请求正常。
201 Created	创建类的请求完全成功。
202 Accepted	已经接受请求，但未处理完成。
204 No Content	请求完全成功，同时HTTP响应不包含响应体。
400 Bad Request	非法请求。 建议直接修改该请求，不要重试该请求。
401 Unauthorized	在客户端提供认证信息后，返回该状态码，表明服务端指出客户端所提供的认证信息不正确或非法。
403 Forbidden	请求被拒绝访问。 返回该状态码，表明请求能够到达服务端，且服务端能够理解用户请求，但是拒绝做更多的事情，因为该请求被设置为拒绝访问，建议直接修改该请求，不要重试该请求。
404 Not Found	所请求的资源不存在。 建议直接修改该请求，不要重试该请求。
405 Method Not Allowed	请求中带有该资源不支持的方法。 建议直接修改该请求，不要重试该请求。
406 Not Acceptable	服务器无法根据客户端请求的内容特性完成请求。

返回值	说明
407 Proxy Authentication Required	请求要求代理的身份认证，与401类似，但请求者应当使用代理进行授权。
408 Request Timeout	服务器等候请求时发生超时。 客户端可以随时再次提交该请求而无需进行任何更改。
409 Conflict	服务器在完成请求时发生冲突。 返回该状态码，表明客户端尝试创建的资源已经存在，或者由于冲突请求的更新操作不能被完成。
500 Internal Server Error	表明服务端能被请求访问到，但是不能理解用户的请求。
501 Not Implemented	服务器不支持请求的功能，无法完成请求。
502 Bad Gateway	充当网关或代理的服务器，从远端服务器接收到了一个无效的请求。
503 Service Unavailable	被请求的服务无效。 建议直接修改该请求，不要重试该请求。
504 Gateway Timeout	网关超时。

19.2 错误码

当您调用API时，如果遇到“APIGW”开头的错误码，请参见[API网关错误码](#)进行处理。

状态码	错误码	错误信息	描述	处理措施
400	VOD.10053	The request parameter is illegal.	请求参数非法，字段填写错误	请检查字段填写是否和接口定义一致
400	VOD.10081	The request parameters are illegal, please check.	请求参数非法，请检查。	请求参数非法，请检查。
400	VST.10001	The request parameter is illegal.	请求参数非法，字段填写错误	请检查字段填写是否和接口定义一致
400	VST.10002	Internal system error.	系统内部一般性错误	请联系技术支持

状态码	错误码	错误信息	描述	处理措施
400	VST.10004	Identity authentication failed.	身份认证失败	请检查Token等认证参数是否正确
403	VOD.10051	Internal system error.	系统内部一般性错误	请联系技术支持
403	VOD.10052	The internal communication of the service is abnormal.	服务内部通信异常	请联系技术支持
403	VOD.10054	Identity authentication failed.	身份认证失败	请检查Token等认证参数是否正确
403	VOD.10055	The user is not authenticated by real name.	用户在CSB完成未实名认证	请检查是否完成实名认证
403	VOD.10056	The user is in an abnormal state.	用户处于异常状态	请检查用户状态是否正确
403	VOD.10057	Tenant ID verification failed.	租户ID校验失败	请检查租户ID是否正确
403	VOD.10058	The request method is incorrect.	请求方式不正确	请检查请求方式
403	VOD.10059	The requested content type is incorrect.	请求内容类型不正确	请检查请求内容类型
403	VOD.10060	Media asset is not downloaded successfully or not released, please check.	媒资处于未下载成功或者未发布状态, 请检查。	媒资处于未下载成功或者未发布状态, 请检查。
403	VOD.10061	The operation failed, and the current media asset status is not allowed to be updated.	操作失败,当前媒资状态不允许更新。	操作失败,当前媒资状态不允许更新。

状态码	错误码	错误信息	描述	处理措施
403	VOD.10063	Media asset type is not supported at this time.	媒资类型暂不支持。	媒资类型暂不支持。
403	VOD.10065	Media asset classification already exists, please check.	媒资分类已存在，请检查。	媒资分类已存在，请检查。
403	VOD.10066	The media asset classification exceeds the maximum level, please check.	媒资分类超过最大层级，请检查。	媒资分类超过最大层级，请检查。
403	VOD.10067	Media asset classification exceeds the maximum number limit, please check.	媒资分类超过最大个数限制，请检查。	媒资分类超过最大个数限制，请检查。
403	VOD.10069	The topic already exists, please check.	主题已存在，请检查。	主题已存在，请检查。
403	VOD.10070	Set message notification failed, no permission to post message to topic, please check.	设置消息通知失败，无权限发布消息到主题，请检查。	设置消息通知失败，无权限发布消息到主题，请检查。
403	VOD.10071	Referer header field verification failed, please check.	Referer头域校验失败，请检查。	Referer头域校验失败，请检查。

状态码	错误码	错误信息	描述	处理措施
403	VOD.10072	The url authentication setting failed. The old key has not expired. The new key cannot be set. Please set the new key again after expiration.	url鉴权设置失败，旧密钥尚未过期，无法设置新密钥，过期后请再次设置新密钥。	url鉴权设置失败，旧密钥尚未过期，无法设置新密钥，过期后请再次设置新密钥。
403	VOD.10073	Failed to create a media asset transcoding task. Contact technical support.	创建媒资转码任务失败，请联系技术支持。	创建媒资转码任务失败，请联系技术支持。
403	VOD.10074	There is no permission to perform this operation.	无权限执行该操作。	无权限执行该操作。
403	VOD.10076	Failed to get the object storage source file.	获取对象存储源文件失败。	获取对象存储源文件失败。
403	VOD.10077	hms request playback interface authentication failed.	hms请求播放接口鉴权失败。	hms请求播放接口鉴权失败。
403	VOD.10078	The public measurement limit, the usage exceeds the threshold.	公测限额，用量超过阈值。	公测限额，用量超过阈值。
403	VOD.10079	The task was processed successfully.	任务处理成功。	任务处理成功。
403	VOD.10080	The task is being processed.	任务处理中。	任务处理中。

状态码	错误码	错误信息	描述	处理措施
403	VOD.10082	The template group already exists.	模板组已存在。	模板组已存在。
403	VOD.10083	The current media asset status does not support this operation.	当前媒资或资源状态不支持该操作。	当前媒资或资源状态不支持该操作。
403	VOD.10084	Media asset release failed.	媒资发布失败。	媒资发布失败。
403	VOD.10085	The number of domain names allowed to be created cannot exceed {0}.	允许创建的域名数不能超过{0}个。	允许创建的域名数不能超过{0}个。
403	VOD.10086	The total number of preheating urls cannot exceed {0}.	预热url总数不能超过{0}个。	预热url总数不能超过{0}个。
403	VOD.10087	The total number of refresh urls cannot exceed {0}.	刷新url总数不能超过{0}个。	刷新url总数不能超过{0}个。
403	VOD.10088	Only operate enabled domain name.	只能操作已启用的域名。	只能操作已启用的域名。
403	VOD.10090	The modification failed, request to confirm the watermark image upload first.	修改失败，请求先确认水印图片上传	修改失败，请求先确认水印图片上传
403	VOD.10091	Media asset transcoding was successful.	媒资转码成功。	媒资转码成功。

状态码	错误码	错误信息	描述	处理措施
403	VOD.10092	Media asset transcoding failed.	媒资转码失败。	媒资转码失败。
403	VOD.10093	The OBS transfer media asset was successfully released.	OBS转存媒资发布成功。	OBS转存媒资发布成功。
403	VOD.10094	The OBS transfer media asset failed to be released.	OBS转存媒资发布失败。	OBS转存媒资发布失败。
403	VOD.10095	The source domain name cannot be configured to speed up the domain name.	源站域名不能配置为加速域名。	源站域名不能配置为加速域名。
403	VOD.10096	The domain name has been used.	该域名已被使用。	该域名已被使用。
403	VOD.10097	Default accelerated domain name cannot be created.	默认加速域名不能进行创建。	默认加速域名不能进行创建。
403	VOD.10098	Operation failed, domain name is configuring.	操作失败，域名配置中。	操作失败，域名配置中。
403	VOD.10099	Media asset is BLOCKED.	媒资处于屏蔽状态	媒资处于屏蔽状态
403	VOD.10100	No key URL has been configured.	未配置获取密钥URL。	未配置获取密钥URL。
403	VOD.10101	Unable to get the key because there is no encryption.	媒资未加密，无法获取密钥。	媒资未加密，无法获取密钥。

状态码	错误码	错误信息	描述	处理措施
403	VOD.10102	Edit task only support Mp4/Flv video.	剪辑任务当前只支持Mp4和Flv片源	剪辑任务当前只支持Mp4和Flv片源
403	VOD.10103	Concat index asset must in input asset list.	参考视频必须是待拼接视频列表中的一员	参考视频必须是待拼接视频列表中的一员
403	VOD.10104	The thumbnail task failed.	截图任务失败	截图任务失败
403	VOD.10105	The review task failed.	审核任务失败	审核任务失败
403	VOD.10106	Cannot delete the default assigned domain name or default domain name.	不能删除默认分配域名或者默认域名。	不能删除默认分配域名或者默认域名。
403	VOD.10107	Default allocation of domain names cannot configure Https.	VOD默认分配域名不能配置Https。	VOD默认分配域名不能配置Https。
403	VOD.10108	The interval of Cut task is too small.	剪切间隔时长过小	剪切间隔时长过小
403	VOD.10109	TinyAsset preheat url is over 10 limit.	最多预热10个短视频指定URL	最多预热10个短视频指定URL
403	VOD.10110	Unable to create cdn preHeating task, please check if the asset is enabled for cdn.	无法创建cdn预热任务, 请检查该媒资是否启用cdn加速域名	无法创建cdn预热任务, 请检查该媒资是否启用cdn加速域名
403	VOD.10111	No access to resources.	无权限访问资源	无权限访问资源

状态码	错误码	错误信息	描述	处理措施
403	VOD.10113	OBS notify config is not set.	OBS事件通知未配置	OBS事件通知未配置
403	VOD.10114	Bucket not Authorized.	桶未授权	桶未授权
403	VOD.10115	The output bucket can not be cannel auth.	已设置为输出的桶无法取消授权	已设置为输出的桶无法取消授权
403	VOD.10116	Event notification is being configured for OBS.	正在为OBS服务配置事件通知。	正在为OBS服务配置事件通知。
403	VOD.10118	Bucket asset is handling..	该桶的音视频同步中	该桶的音视频同步中
403	VOD.10119	URL mapping already exists.	URL映射已经存在	URL映射已经存在
403	VOD.10120	You have arrears, please recharge.	您已欠费，请尽快充值	您已欠费，请尽快充值
403	VOD.10121	URL pull asset failed.	URL拉取媒资失败	URL拉取媒资失败
403	VOD.10122	Cancel url pull failed.	取消URL拉取失败	取消URL拉取失败
403	VOD.10123	The url pull task is processing or stopping can not be delete.	不能删除处理中或取消中的URL拉取任务	不能删除处理中或取消中的URL拉取任务
403	VOD.10124	Watermark template is valid or not exists, {0}	水印模板组无效或不存在，{0}	水印模板组无效或不存在，{0}
403	VOD.10125	Watermark number reach limitation: {0}	水印数量达到上限：{0}	水印数量达到上限：{0}

状态码	错误码	错误信息	描述	处理措施
403	VOD.10126	Uploading confirmation is not allowed in the current media asset status. Please check the media asset status.	媒资当前状态不允许进行上传确认，请确认媒资状态	媒资当前状态不允许进行上传确认，请确认媒资状态
403	VOD.10127	APIGW rate limit.	MAI流控限制	MAI流控限制
403	VOD.10128	The task status does not support recovery operations.	任务状态不支持恢复操作	任务状态不支持恢复操作
403	VOD.10130	The domain name does not support configuration key anti-theft chain: {0}.	域名不支持配置防盗链: {0}	域名不支持配置防盗链: {0}
403	VOD.10131	The watermark template already exists.	水印模板已经存在。	水印模板已经存在。
403	VOD.10132	Account cancellation or public security freeze, access to services is not allowed.	账户注销或公安冻结，不允许访问服务。	账户注销或公安冻结，不允许访问服务。
403	VOD.10133	No permission to operate related resource.	无权限操作相关的资源。	无权限操作相关的资源。
403	VOD.10134	The domain name is invalid and cannot be created.	域名无效，无法创建。	域名无效，无法创建。

状态码	错误码	错误信息	描述	处理措施
403	VOD.10135	The asset is not in the upload state and cannot obtain temporary authorization.	媒资没有处于上传中状态，无法获取临时授权	媒资没有处于上传中状态，无法获取临时授权
403	VOD.10136	This feature is temporarily offline.	该功能临时下线。	该功能临时下线。
403	VOD.10137	The number of published or unpublished meta resources exceeds the threshold limit:{0}	发布或取消发布媒资个数超过阈值限制： {0}	发布或取消发布媒资个数超过阈值限制： {0}
403	VOD.10138	You do not have permission to operate, contact a tenant administrator barrel authorize or sub-account to the tenant administrator privileges conferred OBS.	您没有操作权限，请联系租户管理员进行桶授权或者租户管理员给予账号赋予OBS权限	您没有操作权限，请联系租户管理员进行桶授权或者租户管理员给予账号赋予OBS权限
404	VOD.10062	Media asset or resource does not exist, please check.	媒资或资源不存在，请检查。	媒资或资源不存在，请检查。
404	VOD.10064	Media asset classification does not exist, please check.	媒资分类不存在，请检查。	媒资分类不存在，请检查。

状态码	错误码	错误信息	描述	处理措施
404	VOD.10068	The theme does not exist, please check.	主题不存在，请检查。	主题不存在，请检查。
404	VOD.10075	The object storage source address or destination address is incorrect. Please check.	对象存储源地址或者目的地址不正确，请检查。	对象存储源地址或者目的地址不正确，请检查。
404	VOD.10089	Accelerated domain name does not exist.	加速域名不存在。	加速域名不存在。
404	VOD.10112	OBS Resource not exists.	OBS桶资源不存在	请检查OBS资源或联系技术支持
404	VOD.10129	Subtitle resource does not exist, please check.	字幕文件未找到，请检查	字幕文件未找到，请检查
404	VOD.10150	Workflow does not exist, please check.	工作流不存在	请检查工作流配置或联系技术支持
404	VOD.10151	TemplateGroup does not exist, please check.	转码模板不存在	请检查转码模板配置或联系技术支持
404	VOD.10152	Domain does not exist, please check.	无可域名	请检查域名配置或联系技术支持
500	VOD.10051	Internal system error.	系统内部一般性错误	请联系技术支持
500	VST.10002	Internal system error.	系统内部一般性错误	请联系技术支持

19.3 获取项目 ID

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。您可以通过以下两种方式获取项目ID。

- [从控制台获取项目ID](#)
- [调用API获取项目ID](#)

从控制台获取项目 ID

步骤1 登录[管理控制台](#)。

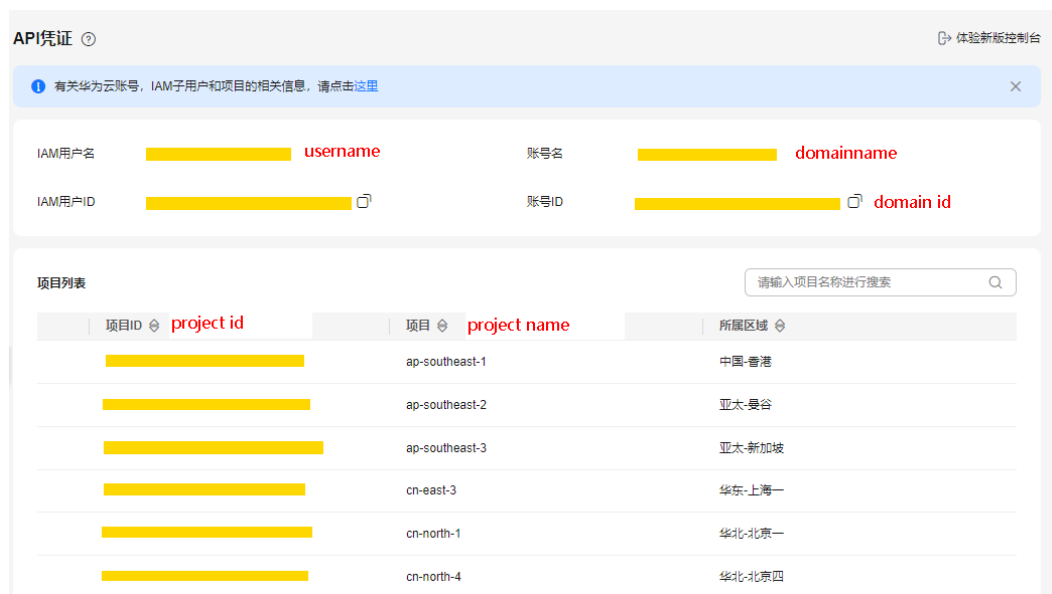
步骤2 鼠标悬停在右上角的用户名，选择下拉列表中“我的凭证”。

图 19-1 管理控制台



步骤3 在“API凭证”页面的项目列表中查看项目ID。

图 19-2 获取项目 ID



----结束

调用 API 获取项目 ID

您还能通过调用[查询指定条件下的项目信息](#)API获取项目ID。

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": "65382450e8f64ac0870cd180d14e684b",
      "is_domain": false,
      "parent_id": "65382450e8f64ac0870cd180d14e684b",
      "name": "region01",
      "description": "",
      "links": {
        "next": null,
        "previous": null,
        "self": "https://www.example.com/v3/projects/a4a5d4098fb4474fa22cd05f897d6b99"
      },
      "id": "a4a5d4098fb4474fa22cd05f897d6b99",
      "enabled": true
    }
  ],
  "links": {
    "next": null,
    "previous": null,
    "self": "https://www.example.com/v3/projects"
  }
}
```

19.4 获取账号 ID

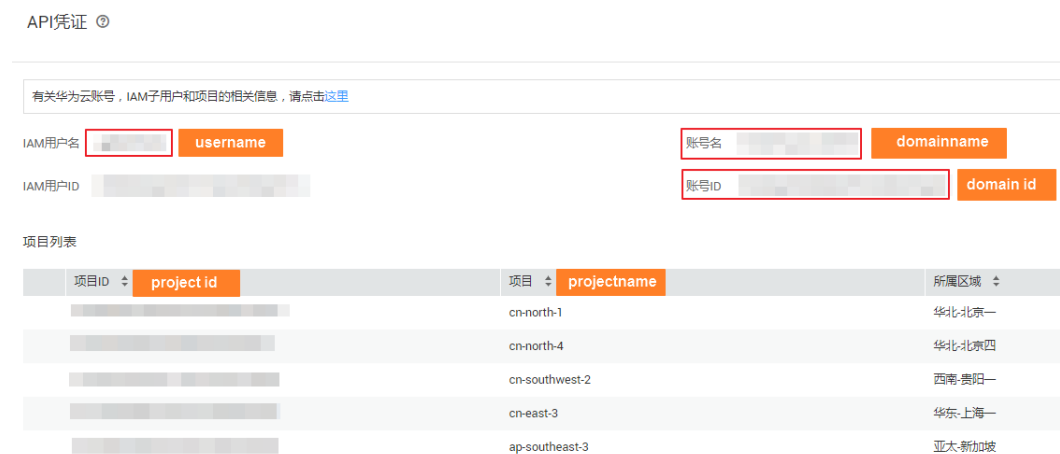
在调用接口的时候，部分URL中需要填入账号ID，所以需要先在管理控制台上获取到账号ID。具体获取步骤如下：

步骤1 登录管理控制台。

步骤2 鼠标悬停在右上角的用户名，选择下拉列表中“我的凭证”。

在“API凭证”页面，即可获取对应的账号ID信息。

图 19-3 获取账号 ID



----结束

19.5 生成 MD5 值

媒资上传和更新

调用[创建媒资：上传方式](#)和[视频更新](#)接口时，可以通过“video_md5”设置媒资文件的MD5值。设置后，OBS会对媒资的MD5值进行检验，具体可参考[设置对象属性](#)。

该MD5值是经过标准的MD5哈希算法计算后，再进行base64编码的。示例代码如下所示：

```
import org.apache.commons.codec.binary.Base64;

import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;

import java.security.DigestInputStream;
import java.security.MessageDigest;
import java.security.NoSuchAlgorithmException;

public class VodDemoObsCheckMd5 {
    public String md5Generate4ObsCheck(String fileUrl)
        throws IOException, NoSuchAlgorithmException {
        String md5Content = null;

        if ((fileUrl != null) && (fileUrl.length() != 0)) {
            File file = new File(fileUrl);

            if (!file.exists()) {
                System.out.println("文件不存在");
            } else if (file.isDirectory()) {
                System.out.println(file.getCanonicalPath() + "是一个目录,无法计算");
            } else {
                FileInputStream fis = new FileInputStream(file);
                DigestInputStream dis = new DigestInputStream(fis,
                    MessageDigest.getInstance("MD5"));
                byte[] buffer = new byte[8192];

                while (dis.read(buffer) > 0) {
                }

                md5Content = new String(Base64.encodeBase64(
                    dis.getMessageDigest().digest()));
                fis.getChannel().position(0L);
                System.out.println("该文件MD5为: " + md5Content);
                fis.close();
            }
        } else {
            System.out.println("缺少文件名");
        }

        return md5Content;
    }
}
```

上传校验

调用[上传检验](#)接口时，点播服务会根据媒资的MD5值来检查是否已有重复的媒资文件。MD5值的生成方式是取媒资文件的1024字节，并进行MD5计算，示例代码如下所示：

```
import java.io.File;
import java.io.IOException;
```



```
import java.nio.ByteBuffer;
import java.nio.channels.SeekableByteChannel;
import java.nio.file.Files;
import java.nio.file.Path;
import java.nio.file.Paths;
import java.nio.file.StandardOpenOption;
import org.apache.commons.codec.digest.DigestUtils;

public class VodDemoDuplicateCheckMd5 {
    public static String computeMd5ByFile(String fileUrl) {
        String md5Content = null;
        Path targetFile = Paths.get(fileUrl);
        try (SeekableByteChannel channel = Files.newByteChannel(targetFile, StandardOpenOption.READ)) {
            ByteBuffer byteBuffer = ByteBuffer.allocate(1025);
            channel.read(byteBuffer);
            byteBuffer.flip();
            byte[] data = new byte[byteBuffer.limit()];
            byteBuffer.get(data);
            md5Content = DigestUtils.md5Hex(data);
        } catch (IOException e) {
            throw new RuntimeException(String.format("Read file %s failed.", fileUrl));
        }
        return md5Content;
    }
}
```

19.6 日志管理

日志说明

- **日志文件延迟时间：**您可以在日志管理中查询到6小时之前的日志文件。
- **日志命名规则：**日志时间跨度-加速域名-服务范围.gz，其中服务范围：“cn”代表中国大陆，“ov”代表中国大陆境外。如“2018021123-www.example01.com-ov.gz”。
- **日志生成规则：**每个加速域名日志默认按小时分割，每天24个日志文件。
- **日志内容示例：**

```
[05/Feb/2018:07:54:52 +0800] x.x.x.x 1 "-" "HTTP/1.1" "GET" "www.test.com" "/test/1234.apk" 206
720 HIT "Mozilla/5.0 (Linux; U; Android 6.0; zh-cn; EVA-AL10 Build/HUAWEIEVA-AL10) AppleWebKit/
533.1 (KHTML, like Gecko) Mobile Safari/533.1" "bytes=-256"
```

各字段从左到右含义如表19-2所示。

表 19-2 日志字段说明

序号	字段含义	字段示例
1	日志生成时间	[05/Feb/2018:07:54:52 +0800]
2	访问IP地址	x.x.x.x
3	响应时间（单位ms）	1
4	Referer信息	-
5	HTTP协议标识	HTTP/1.1
6	HTTP请求方式	GET
7	加速域名	www.test.com

序号	字段含义	字段示例
8	请求路径	/test/1234.apk
9	HTTP状态码	206
10	返回字节数大小	720
11	缓存命中状态	HIT
12	User-Agent信息，其作用是让服务器能够识别用户使用的操作系统及版本、CPU类型、浏览器及版本信息等。	Mozilla/5.0 (Linux; U; Android 6.0; zh-cn; EVA-AL10 Build/HUAWEIEVA-AL10) AppleWebKit/533.1 (KHTML, like Gecko) Mobile Safari/533.1
13	Range信息，其作用是在HTTP请求头中指定返回数据的范围，即第一个字节的位置和最后一个字节的位置。 bytes参数值表示方法一般分为如下三类： <ul style="list-style-type: none"> • bytes=x-y：表示请求第x个字节到第y个字节的数据内容。 • bytes=-y：表示请求最后y个字节的数据内容。 • bytes=x-：表示请求第x字节到最后一个字节的数据内容。 	bytes=-256

19.7 分段上传代码示例

19.7.1 Java

分段上传Java语言的示例代码，如下所示：

```

/*
 * Copyright (c) Huawei Technologies Co., Ltd. 2019-2022. All rights reserved.
 */
package com.huaweicloud.vod;

import com.huaweicloud.sdk.core.auth.BasicCredentials;
import com.huaweicloud.sdk.core.auth.ICredential;
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.vod.v1.VodClient;
import com.huaweicloud.sdk.vod.v1.model.ConfirmAssetUploadReq;
import com.huaweicloud.sdk.vod.v1.model.ConfirmAssetUploadRequest;
import com.huaweicloud.sdk.vod.v1.model.ConfirmAssetUploadResponse;
import com.huaweicloud.sdk.vod.v1.model.CreateAssetByFileUploadReq;

```

```

import com.huaweicloud.sdk.vod.v1.model.CreateAssetByFileUploadRequest;
import com.huaweicloud.sdk.vod.v1.model.CreateAssetByFileUploadResponse;
import com.huaweicloud.sdk.vod.v1.model.ShowAssetTempAuthorityRequest;
import com.huaweicloud.sdk.vod.v1.model.ShowAssetTempAuthorityResponse;
import com.huaweicloud.sdk.vod.v1.region.VodRegion;

import cn.hutool.http.HttpRequest;
import cn.hutool.http.HttpResponse;

import org.apache.commons.codec.binary.Base64;
import org.apache.commons.lang3.StringUtils;
import org.dom4j.Document;
import org.dom4j.DocumentException;
import org.dom4j.DocumentHelper;
import org.dom4j.Element;

import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
import java.security.MessageDigest;
import java.time.LocalDateTime;
import java.time.format.DateTimeFormatter;
import java.util.Arrays;
import java.util.List;
import java.util.TimeZone;

/**
 * 分段上传示例
 */
public class PartUploadDemo {

    // 设置缓冲区大小，每次读取文件分段的大小，根据情况自行设定
    public static final int bufferSize = 1024 * 1024; // 1MB

    // 区域
    public static final String REGION_NORTH4 = "cn-north-4";
    public static final String REGION_NORTH1 = "cn-north-1";
    public static final String REGION_EAST2 = "cn-east-2";

    // ak/sk
    private static final String AK = "";
    private static final String SK = "";

    public static void main(String[] args) {
        // 本地要上传的媒资路径
        String filePath = "";

        PartUploadDemo uploadDemo = new PartUploadDemo();
        // 上传媒资文件
        uploadDemo.uploadPartFile(filePath);
    }

    /**
     * 分段上传
     * @param filePath 上传文件的本地路径
     */
    public void uploadPartFile(String filePath) {
        // 校验一下文件路径和文件
        File file = validFile(filePath);
        try {
            String fileName = file.getName();

            // 此处仅以MP4文件示例，其他格式可参考官网说明
            String fileType = "MP4";
            String fileContentType = "video/mp4";

            // FileInfo fileInfo = new FileInfo(filePath, file);
            // 1.初始化鉴权并获取vodClient
            VodClient vodClient = this.createVodClient();

```

```

System.out.println("开始创建媒资:" + file.getName());
// 2.创建点播媒资
CreateAssetByFileUploadReq reqbody = this.buildFileUploadReq(fileName, fileType, null, null);
CreateAssetByFileUploadResponse assetResponse = this.createAssetByFileUpload(vodClient,
reqbody);
// 3.获取初始化上传任务授权
ShowAssetTempAuthorityResponse initAuthResponse = this.initPartUploadAuthority(vodClient,
assetResponse, fileType);
// 4.初始化上传任务
String uploadId = this.initPartUpload(initAuthResponse.getSignStr(), fileType);

// 文件分段计数
int partNumber = 1;
// 缓冲区
byte[] fileByte = new byte[bufferSize];
// 记录读取的长度
int readLength = 0;

// 7.读取文件内容, 循环5-6步上传所有分段
FileInputStream fis = new FileInputStream(file);
// MD5
MessageDigest md = MessageDigest.getInstance("MD5");
while ((readLength = fis.read(fileByte)) > 0) {
// 读取的长度小于缓冲区长度, 复制有效长度内数据, 用于最后一段
if(readLength < bufferSize) {
fileByte = Arrays.copyOf(fileByte, readLength);
}
// 先md5,再base64 encode
byte[] digest = md.digest(fileByte);
String contentMd5 = new String(Base64.encodeBase64(digest));
System.out.println("该文件第"+(partNumber)+"段MD5为: " + contentMd5);

// 5.获取上传分段的授权
ShowAssetTempAuthorityResponse partUploadAuthorityResponse =
this.getPartUploadAuthority(vodClient, fileType, assetResponse, contentMd5, uploadId, partNumber);

// 6.上传分段
this.uploadPartFile(partUploadAuthorityResponse.getSignStr(), fileByte, contentMd5);

// 段号自增
partNumber++;
}

fis.close();

// 8.获取已上传分段的授权
ShowAssetTempAuthorityResponse listPartUploadAuthorityResponse =
this.listUploadedPartAuthority(vodClient, assetResponse, uploadId);
// 9.获取已上传的分段
String partInfo = this.listUploadedPart(listPartUploadAuthorityResponse.getSignStr());
// 10.获取合并段授权
ShowAssetTempAuthorityResponse mergePartUploadAuthorityResponse =
this.mergeUploadedPartAuthority(vodClient, assetResponse, uploadId);
// 11.合并上传分段
this.mergeUploadedPart(mergePartUploadAuthorityResponse.getSignStr(), partInfo);
// 12.确认媒资上传
this.confirmUploaded(vodClient, assetResponse);
System.out.println("创建媒资结束 assetId:" + assetResponse.getAssetId());
} catch (ServiceResponseException e) {
e.printStackTrace();
System.out.println(String.valueOf(e.getStatusCode()));
System.out.println(e.getRequestId());
System.out.println(e.getErrorCode());
System.out.println(e.getErrorMsg());
} catch (Exception e) {
e.printStackTrace();
}
}
}

```

```

/**
 * 校验文件路径和文件
 * @param filePath
 */
private File validFile(String filePath){
    if (StringUtils.isEmpty(filePath)) {
        throw new RuntimeException("输入的文件路径为空! ");
    }
    File file = new File(filePath);
    if (!file.exists()) {
        throw new RuntimeException("文件不存在! ");
    } else if (file.isDirectory()) {
        try {
            throw new RuntimeException(file.getCanonicalPath() + "这是一个目录! ");
        } catch (IOException e) {
            throw new RuntimeException(e);
        }
    }
    return file;
}

/**
 * 1.构建鉴权
 * @return
 */
public VodClient createVodClient(){
    ICredential auth = new BasicCredentials()
        .withAk(AK)
        .withSk(SK)
        ;
    return VodClient.newBuilder()
        .withCredential(auth)
        .withRegion(VodRegion.valueOf(REGION_NORTH4))
        .build();
}

/**
 * 2.创建媒资
 * @param client
 * @param reqbody
 * @return
 */
public CreateAssetByFileUploadResponse createAssetByFileUpload(VodClient client,
CreateAssetByFileUploadReq reqbody){
    System.out.println("createAssetByFileUpload start");
    CreateAssetByFileUploadRequest request = new CreateAssetByFileUploadRequest();
    // 设置 X-Sdk-Date参数, ak/sk认证必传
    request.setXSdkDate(getXSdkDate());
    // 设置上传参数
    request.withBody(reqbody);

    // 调用创建媒资
    CreateAssetByFileUploadResponse response = client.createAssetByFileUpload(request);
    System.out.println("createAssetByFileUpload end; createAssetResponse:" + response.toString());
    return response;
}

/**
 * 构建创建媒资请求参数
 * @param fileName
 * @param videoName
 * @param title
 * @return
 */
public CreateAssetByFileUploadReq buildFileUploadReq(String fileName, String videoType, String
videoName, String title){
    CreateAssetByFileUploadReq req = new CreateAssetByFileUploadReq();
    req.withVideoName(StringUtils.isNotEmpty(videoName) ? videoName : fileName);
}

```

```

req.withTitle(StringUtils.isEmpty(title) ? title : fileName);
// 设置媒资类型
req.withVideoType(videoType);
return req;
}

/**
 * 取 X-Sdk-Date 参数, 当前UTC时间, 时间格式 yyyyHHddTHHmssZ 例如: 20240312T092514Z
 * @return
 */
public String getXSdkDate(){
    TimeZone zone = TimeZone.getTimeZone("UTC");
    DateTimeFormatter formatter = DateTimeFormatter.ofPattern("yyyyMMdd'T'HHmss'Z");
    return LocalDateTime.now(zone.toZoneId()).format(formatter);
}

/**
 * 3.获取初始化上传任务授权
 * @param client
 * @param assetResponse
 */
public ShowAssetTempAuthorityResponse initPartUploadAuthority(VodClient client,
CreateAssetByFileUploadResponse assetResponse, String fileContentType){
    System.out.println("获取初始化上传任务授权 initPartUploadAuthority start");
    ShowAssetTempAuthorityRequest request = new ShowAssetTempAuthorityRequest();
    request.setXSdkDate(getXSdkDate());
    // 设置参数
    request.withHttpVerb("POST");
    request.withBucket(assetResponse.getTarget().getBucket());
    request.withObjectKey(assetResponse.getTarget().getObject());
    request.withContentType(fileContentType);
    // 发送初始化请求
    ShowAssetTempAuthorityResponse response = client.showAssetTempAuthority(request);
    System.out.println("获取初始化上传任务授权 initPartUploadAuthority end; response: " +
response.toString());
    return response;
}

/**
 * 4.初始化分段上传
 * @param signStr
 * @param contentType
 * @return
 */
public String initPartUpload(String signStr, String contentType) throws DocumentException {
    System.out.println("初始化分段上传 initPartUpload start");
    HttpResponse response = HttpRequest.post(signStr).header("Content-type",contentType).execute();
    System.out.println(response.body());
    Document document = DocumentHelper.parseText(response.body());
    Element root = document.getRootElement();
    Element u = root.element("UploadId");
    System.out.println("初始化分段上传 initPartUpload end; UploadId:" + u.getText());
    return u.getText();
}

/**
 * 5.获取分段上传授权
 * @param client
 * @param fileContentType
 * @param assetResponse
 * @param contentMd5
 * @param uploadId
 * @param partNumber
 * @return
 */
public ShowAssetTempAuthorityResponse getPartUploadAuthority(VodClient client, String
fileContentType, CreateAssetByFileUploadResponse assetResponse, String contentMd5, String uploadId, int
partNumber) {
    System.out.println("获取分段上传授权 getPartUploadAuthority start; partNumber: " + partNumber);

```

```

ShowAssetTempAuthorityRequest request = new ShowAssetTempAuthorityRequest();
request.setXSdkDate(getXSdkDate());
request.withHttpVerb("PUT");
request.withBucket(assetResponse.getTarget().getBucket());
request.withObjectKey(assetResponse.getTarget().getObject());
request.withContentType(fileContentType);
request.withContentMd5(contentMd5);
request.withUploadId(uploadId);
request.withPartNumber(partNumber);
ShowAssetTempAuthorityResponse response = client.showAssetTempAuthority(request);
System.out.println("获取分段上传授权 getPartUploadAuthority end; partNumber: " + partNumber + ";
response" + response.toString());
return response;
}

/**
 * 6.上传分段
 * @param signStr
 * @param fileByte
 * @param contentMd5
 */
public void uploadPartFile(String signStr, byte[] fileByte, String contentMd5) {
System.out.println("上传分段 uploadPartFile start");
HttpResponse response = HttpRequest.put(signStr)
// 此处contentMd5不需要转义
.header("Content-MD5", contentMd5)
.header("Content-Type", "application/octet-stream")
.body(fileByte).execute();
System.out.println(response.toString());
if (response.getStatus() != 200) {
throw new RuntimeException("上传分段 uploadPartFile end; 上传失败! ");
}
System.out.println("上传分段 uploadPartFile end");
}

/**
 * 8.获取列举已上传段的授权
 * @param client
 * @param assetResponse
 * @param uploadId
 * @return
 */
public ShowAssetTempAuthorityResponse listUploadedPartAuthority(VodClient client,
CreateAssetByFileUploadResponse assetResponse, String uploadId){
System.out.println("获取列举已上传段的授权 listUploadedPartAuthority start");
ShowAssetTempAuthorityRequest request = new ShowAssetTempAuthorityRequest();
request.setXSdkDate(getXSdkDate());
request.withHttpVerb("GET");
request.withBucket(assetResponse.getTarget().getBucket());
request.withObjectKey(assetResponse.getTarget().getObject());
request.withUploadId(uploadId);
ShowAssetTempAuthorityResponse response = client.showAssetTempAuthority(request);
System.out.println("获取列举已上传段的授权 listUploadedPartAuthority end; response: " +
response.toString());
return response;
}

/**
 * 9.查询已上传的分段
 * @param signStr
 * @return
 */
public String listUploadedPart(String signStr) throws DocumentException {
System.out.println("查询已上传的分段 listUploadedPart start");
int partNumberMarker = 0;
Element mergerRoot = DocumentHelper.createElement("CompleteMultipartUpload");
while(true) {
//列举分段
HttpResponse response = HttpRequest.get(signStr + "&part-number-marker=" +

```

```

partNumberMarker).execute();
    System.out.println("listUploadedPartResponse:" + response.body());
    Document responseDocument = DocumentHelper.parseText(response.body());
    Element rootResponse = responseDocument.getRootElement();
    List<Element> elementsResponse = rootResponse.elements("Part");
    Element partNumberMarkerElement = rootResponse.element("NextPartNumberMarker");
    for (Element e : elementsResponse) {
        Element te = DocumentHelper.createElement("Part");
        te.add(e.element("PartNumber").createCopy());
        te.add(e.element("ETag").createCopy());
        mergerRoot.add(te);
    }
    partNumberMarker = Integer.valueOf(partNumberMarkerElement.getText());
    if(partNumberMarker % 1000 != 0) {
        break;
    }
}
System.out.println(mergerRoot.asXML());
System.out.println("查询已上传的分段 listUploadedPart end");
return mergerRoot.asXML();
}

/**
 * 10.获取合并段授权
 * @param client
 * @param assetResponse
 * @param uploadId
 * @return
 */
public ShowAssetTempAuthorityResponse mergeUploadedPartAuthority(VodClient client,
CreateAssetByFileUploadResponse assetResponse, String uploadId) {
    System.out.println("获取合并段授权 mergeUploadedPartAuthority start");
    ShowAssetTempAuthorityRequest request = new ShowAssetTempAuthorityRequest();
    request.setXSdkDate(getXSdkDate());
    request.withHttpVerb("POST");
    request.withBucket(assetResponse.getTarget().getBucket());
    request.withObjectKey(assetResponse.getTarget().getObject());
    request.withUploadId(uploadId);
    ShowAssetTempAuthorityResponse response = client.showAssetTempAuthority(request);
    System.out.println("获取合并段授权 mergeUploadedPartAuthority end; response: " +
response.toString());
    return response;
}

/**
 * 11.合并分段
 */
public void mergeUploadedPart(String signStr, String partInfo) {
    System.out.println("合并分段 mergeUploadedPart start");
    // 请求消息头中增加 "Content-Type", 值设置为 "application/xml" 。
    HttpResponse response = HttpRequest.post(signStr)
        .header("Content-Type","application/xml")
        .body(partInfo)
        .execute();
    System.out.println(response.toString());
    if (response.getStatus() != 200) {
        throw new RuntimeException("合并分段 mergeUploadedPart end; 合并段失败!");
    }
    System.out.println("合并分段 mergeUploadedPart end");
}

/**
 * 12.确认上传完成
 */
public void confirmUploaded(VodClient client, CreateAssetByFileUploadResponse assetResponse) {
    System.out.println("确认上传完成 confirmUploaded start");
    ConfirmAssetUploadRequest request = new ConfirmAssetUploadRequest();
    request.setXSdkDate(getXSdkDate());
    ConfirmAssetUploadReq body = new ConfirmAssetUploadReq();
}

```



```

        body.withStatus(ConfirmAssetUploadReq.StatusEnum.fromValue("CREATED"));
        body.withAssetId(assetResponse.getAssetId());
        request.withBody(body);
        ConfirmAssetUploadResponse response = client.confirmAssetUpload(request);
        System.out.println("上传完成, assetId:" + response.toString());
    }
}

```

19.7.2 POM 文件

pom.xml文件示例，如下所示：

```

<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/
maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>

  <groupId>com.huaweicloud.vod</groupId>
  <artifactId>partUploadDemo</artifactId>
  <version>1.0-SNAPSHOT</version>

  <properties>
    <maven.compiler.source>8</maven.compiler.source>
    <maven.compiler.target>8</maven.compiler.target>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
  </properties>

  <dependencies>
    <dependency>
      <groupId>com.huaweicloud.sdk</groupId>
      <artifactId>huaweicloud-sdk-vod</artifactId>
      <version>3.1.93</version>
    </dependency>
    <dependency>
      <groupId>cn.hutool</groupId>
      <artifactId>hutool-all</artifactId>
      <version>5.8.23</version>
    </dependency>
    <dependency>
      <groupId>org.apache.commons</groupId>
      <artifactId>commons-lang3</artifactId>
      <version>3.9</version>
    </dependency>
    <dependency>
      <groupId>commons-codec</groupId>
      <artifactId>commons-codec</artifactId>
      <version>1.15</version>
    </dependency>
    <dependency>
      <groupId>org.dom4j</groupId>
      <artifactId>dom4j</artifactId>
      <version>2.1.4</version>
    </dependency>
  </dependencies>
</project>

```

19.7.3 JavaScript

分段上传JavaScript语言的示例代码，如下所示：

```

<!DOCTYPE html>
<html>
  <head>
    <script>
      // token认证

```

```

var token = "";
var projectId = "";
const region = "cn-north-4";

var urlList;

function initUrl(projectId){
  let url = {
    // 创建媒资接口
    createAssetUrl : "https://vod."+region+".myhuaweicloud.com/v1.0/"+projectId+"/asset",
    // 获取授权接口
    initAuthUrl : "https://vod."+region+".myhuaweicloud.com/v1.0/"+projectId+"/asset/authority",
    // 确认媒资上传接口
    confirmUploadedUrl : "https://vod."+region+".myhuaweicloud.com/v1.0/"+projectId+"/asset/
status/uploaded"
  }
  return url;
}

// 文件分段大小1Mb，根据用户实际情况自行调整
const bufferSize = 1024 * 1024;

// 开始上传
function startUpload(){
  token = document.getElementById("token").value;
  projectId = document.getElementById("projectId").value;

  urlList = initUrl(projectId);

  let files = document.getElementById("file").files;
  let file = files[0];

  // 此处仅以MP4文件示例，其他格式可参考官网说明
  let fileType = "MP4";
  let fileContentType = "video/mp4";

  // 1. 以token鉴权为例，设置请求头
  let headers = {
    "X-Auth-Token": token,
    "Content-Type": "application/json"
  }

  // 2.创建点播媒资
  let assetRsp = createAsset(file.name, file.name, fileType, headers);

  // 3.获取初始化上传任务授权
  let initAuthResponse = getInitAuth(assetRsp, fileContentType, headers);

  // 4.初始化分段上传任务
  let uploadId = initPartUpload(initAuthResponse, assetRsp, fileContentType);

  // 文件分段计数
  let partNumber = 1;
  // 文件读取位置
  let position = 0;

  let blobSlice = File.prototype.mozSlice || File.prototype.webkitSlice || File.prototype.slice;
  // 7.循环5-6上传分段
  readAndUpload(file, blobSlice, position, partNumber, assetRsp, fileContentType, uploadId,
headers,
  function(assetRsp, uploadId, headers){
    // 8.获取列举已上传段授权
    let listAuthResp = listUploadedPartAuthority(assetRsp, uploadId, headers);
    // 9.列举已上传段
    let partInfo = listUploadedPart(listAuthResp.sign_str, assetRsp, uploadId);
    // 10.获取合并段授权
    let mergeAuthResp = mergeUploadedPartAuthority(assetRsp, uploadId, headers);
    // 11.合并段
  }
}

```

```

        mergeUploadedPart(mergeAuthResp.sign_str, partInfo, assetRsp, uploadId);
        // 12.确认上传
        confirmUploaded(assetRsp.asset_id, headers);
        alert("上传结束; assetId:" + assetRsp.asset_id);
    }
    );
}

// 2. 创建点播媒资
function createAsset(title, videoName, videoType, headers){
    let body = {
        "title":title,
        "video_name":videoName,
        "video_type":videoType
    }
    let resp = sendRequest("POST", urlList.createAssetUrl, JSON.stringify(body), headers);
    return JSON.parse(resp);
}

// 3.获取上传初始化任务授权
function getInitAuth(assetRsp, fileContentType, headers){
    let params = {
        "http_verb" : "POST",
        "content_type" : fileContentType,
        "bucket" : assetRsp.target.bucket,
        "object_key" : assetRsp.target.object
    }
    let temp = "?";
    for(let e in params){
        temp += e + "=" + params[e] + "&";
    }
    let resp = sendRequest("GET", urlList.initAuthUrl + temp, null, headers);
    return JSON.parse(resp);
}

// 4.初始化分段上传, 返回uploadId
function initPartUpload(signStr, assetRsp, contentType){
    let initUrl = "https://" + assetRsp.target.bucket + ".obs." + region + ".myhuaweicloud.com/"
        + assetRsp.target.object + "?uploads&" + signStr.sign_str;
    let resp = sendRequest("POST", initUrl, null, {"Content-Type":contentType});
    let domParser = new DOMParser();
    let dom = domParser.parseFromString(resp, "text/xml");
    return dom.getElementsByTagName("UploadId")[0].firstChild.nodeValue;
}

// 5.获取分段上传授权
function getPartUploadAuthority(assetRsp, fileContentType, uploadId, contentMd5, partNumber,
headers){
    let params = {
        "http_verb" : "PUT",
        "content_type" : fileContentType,
        "bucket" : assetRsp.target.bucket,
        "object_key" : assetRsp.target.object,
        "content_md5" : encodeURIComponent(contentMd5),// 有特殊符号, 此处加转义
        "upload_id" : uploadId,
        "part_number" : partNumber
    }
    let temp = "?";
    for(let e in params){
        temp += e + "=" + params[e] + "&";
    }
    let resp = sendRequest("GET", urlList.initAuthUrl + temp, null, headers);
    return JSON.parse(resp);
}

// 6.上传分段
function uploadPartFile(uploadAuth, assetRsp, contentMd5, partNumber, uploadId, content)
{
    let url = "https://" + assetRsp.target.bucket + ".obs." + region + ".myhuaweicloud.com/"

```

```

        + assetRsp.target.object + "?partNumber="+partNumber+"&uploadId=" + uploadId + "&" +
uploadAuth.sign_str;
    let headers = {
        "Content-Type" : "application/octet-stream",
        "Content-MD5" : contentMd5
    }
    let resp = sendRequest("PUT", url, content, headers);
}

// 7.循环5-6上传分段
function readAndUpload(file, blobSlice, position, partNum, assetRsp, fileContentType, uploadId,
headers, afterUpload){
    let fileReader = new FileReader();
    fileReader.onload = function(e) {
        // 先文件分段md5
        let md5 = md5_min(e.target.result);
        // md5转字节数组
        let bmd5 = Str2Bytes(md5);
        // base64 encode md5字节数组
        let content_md5 = base64js_min.fromByteArray(bmd5);

        // 5.获取分段上传授权
        let uploadAuth = getPartUploadAuthority(assetRsp, fileContentType, uploadId, content_md5,
partNum, headers);
        // 6.上传分段
        uploadPartFile(uploadAuth, assetRsp, content_md5, partNum, uploadId, e.target.result);

        partNum++;

        // 判断是否读完, 未读完继续上传
        if(position + bufferSize < file.size){
            readAndUpload(file, blobSlice, position + bufferSize, partNum, assetRsp, fileContentType,
uploadId, headers, afterUpload);
        } else {
            // 全部上传完成后执行后续步骤
            afterUpload(assetRsp, uploadId, headers);
        }
    }
    fileReader.readAsArrayBuffer(blobSlice.call(file, position, position + bufferSize));
}

// 8.获取列举分段上传授权
function listUploadedPartAuthority(assetRsp, uploadId, headers){
    let params = {
        "http_verb" : "GET",
        "bucket" : assetRsp.target.bucket,
        "object_key" : assetRsp.target.object,
        "upload_id" : uploadId
    }
    let temp = "?";
    for(let e in params){
        temp += e + "=" + params[e] + "&";
    }
    let resp = sendRequest("GET", urlList.initAuthUrl + temp, null, headers);
    return JSON.parse(resp);
}

// 9.查询已上传的分段
function listUploadedPart(signStr, assetRsp, uploadId){
    let url = "https://" + assetRsp.target.bucket + ".obs." + region + ".myhuaweicloud.com/"
+ assetRsp.target.object + "?" + signStr + "&uploadId=" + uploadId;
    let nextPartNumberMarker = 0;

    let result = "<CompleteMultipartUpload>";
    let domParser = new DOMParser();
    while(true) {
        let resp = sendRequest("GET" , url + "&part-number-marker=" + nextPartNumberMarker, null,
{});
        let dom = domParser.parseFromString(resp, "text/xml");
    }
}

```

```

        let part = dom.getElementsByTagName("Part");

        // 构建合并段的参数
        for(let i = 0; i < part.length; i++){
            let ele = part[i];
            let num = ele.getElementsByTagName("PartNumber")[0].firstChild.nodeValue;
            let tag = ele.getElementsByTagName("ETag")[0].firstChild.nodeValue;
            result += "<Part>" +
                "<PartNumber>" + num + "</PartNumber>" +
                "<ETag>" + tag + "</ETag>" +
                "</Part>"
            ;
        }

        nextPartNumberMarker = Number(dom.getElementsByTagName("NextPartNumberMarker")
[0].firstChild.nodeValue);

        if(nextPartNumberMarker % 1000 != 0) {
            break;
        }
    }
    result += "</CompleteMultipartUpload>";
    return result;
}

// 10.获取合并段授权
function mergeUploadedPartAuthority(assetRsp, uploadId, headers){
    let params = {
        "http_verb"    : "POST",
        "bucket"       : assetRsp.target.bucket,
        "object_key"   : assetRsp.target.object,
        "upload_id"    : uploadId
    }
    let temp = "?";
    for(let e in params){
        temp += e + "=" + params[e] + "&";
    }
    let resp = sendRequest("GET", urlList.initAuthUrl + temp, null, headers);
    return JSON.parse(resp);
}

// 11.合并段
function mergeUploadedPart(signStr, partInfo, assetRsp, uploadId){
    let url = "https://" + assetRsp.target.bucket + ".obs." + region + ".myhuaweicloud.com/"
        + assetRsp.target.object + "?" + signStr + "&uploadId=" + uploadId;
    let resp = sendRequest("POST" , url, partInfo, {"Content-Type":"application/xml"});
}

// 12.确认上传
function confirmUploaded(assetId, headers){
    let body = {
        "asset_id": assetId,
        "status":"CREATED"
    };
    let resp = sendRequest("POST", urlList.confirmUploadedUrl, JSON.stringify(body), headers);
    return console.log(resp);
}

// 发送请求并返回结果，可以实际框架情况进行替换
function sendRequest(method, url, data, headers){
    var xhr = new XMLHttpRequest();
    xhr.open(method, url, false);
    for(let i in headers){
        xhr.setRequestHeader(i, headers[i]);
    }
    xhr.send(data);
    return xhr.responseText;
}

```

```

// base64, 可以根据实际使用的前端框架进行替换
var base64js_min = createCommonjsModule(function(e, t) {
  e.exports = function o(n, i, a) {
    function s(t, e) {
      if (!i[t]) {
        if (!n[t]) {
          var r = "function" == typeof commonjsRequire && commonjsRequire;
          if (!e && r)
            return r(t, !0);
          if (!l)
            return l(t, !0);
          throw (e = new Error("Cannot find module '" + t + "'")).code =
"MODULE_NOT_FOUND",
            e
        }
        r = i[t] = {
          exports: {}
        },
        n[t][0].call(r.exports, function(e) {
          return s(n[t][1][e] || e)
        }, r, r.exports, o, n, i, a)
      }
      return i[t].exports
    }
    for (var l = "function" == typeof commonjsRequire && commonjsRequire, e = 0; e < a.length; e
++)
      s(a[e]);
    return s
  }({
    "/": [function(e, t, r) {
      r.byteLength = function(e) {
        var e = d(e)
          , t = e[0]
          , e = e[1];
        return 3 * (t + e) / 4 - e
      }
      ,
      r.toByteArray = function(e) {
        for (var t, r = d(e), o = r[0], r = r[1], n = new u(function(e, t) {
          return 3 * (e + t) / 4 - t
        })(o, r), i = 0, a = 0 < r ? o - 4 : o, s = 0; s < a; s += 4)
          t = l[e.charCodeAt(s)] << 18 | l[e.charCodeAt(s + 1)] << 12 | l[e.charCodeAt(s + 2)] <<
6 | l[e.charCodeAt(s + 3)],
          n[i++] = t >> 16 & 255,
          n[i++] = t >> 8 & 255,
          n[i++] = 255 & t;
          2 === r && (t = l[e.charCodeAt(s)] << 2 | l[e.charCodeAt(s + 1)] >> 4,
          n[i++] = 255 & t);
          1 === r && (t = l[e.charCodeAt(s)] << 10 | l[e.charCodeAt(s + 1)] << 4 | l[e.charCodeAt(s
+ 2)] >> 2,
          n[i++] = t >> 8 & 255,
          n[i++] = 255 & t);
          return n
        }
      ,
      r.fromByteArray = function(e) {
        for (var t, r = e.length, o = r % 3, n = [], i = 0, a = r - o; i < a; i += 16383)
          n.push(function(e, t, r) {
            for (var o, n = [], i = t; i < r; i += 3)
              o = (e[i] << 16 & 16711680) + (e[i + 1] << 8 & 65280) + (255 & e[i + 2]),
              n.push(function(e) {
                return s[e >> 18 & 63] + s[e >> 12 & 63] + s[e >> 6 & 63] + s[63 & e]
              })(o);
            return n.join("")
          })(e, i, a < i + 16383 ? a : i + 16383);
          1 == o ? (t = e[r - 1],
          n.push(s[t >> 2] + s[t << 4 & 63] + "==")) : 2 == o && (t = (e[r - 2] << 8) + e[r - 1],
          n.push(s[t >> 10] + s[t >> 4 & 63] + s[t << 2 & 63] + "=="));
          return n.join("")
        }
      ]
    }
  })
}

```

```

    }
    ;
    for (var s = [], l = [], u = "undefined" != typeof Uint8Array ? Uint8Array : Array, o =
"ABCDEFGHIIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/", n = 0, i = o.length; n < i; +
+n)
        s[n] = o[n],
        l[o.charCodeAt(n)] = n;
    function d(e) {
        var t = e.length;
        if (0 < t % 4)
            throw new Error("Invalid string. Length must be a multiple of 4");
        e = e.indexOf("="),
        t = (e = -1 === e ? t : e) === t ? 0 : 4 - e % 4;
        return [e, t]
    }
    l["-".charCodeAt(0)] = 62,
    l["_".charCodeAt(0)] = 63
    }, {}]
    }, {}, [])("/")
    })
    // md5, 可以根据实际使用的前端框架进行替换
    , md5_min = createCommonjsModule(function(module) {
        !function() {
            function t(e) {
                e ? (d[0] = d[16] = d[1] = d[2] = d[3] = d[4] = d[5] = d[6] = d[7] = d[8] = d[9] = d[10] =
d[11] = d[12] = d[13] = d[14] = d[15] = 0,
                this.blocks = d,
                this.buffer8 = l) : a ? (e = new ArrayBuffer(68),
                this.buffer8 = new Uint8Array(e),
                this.blocks = new Uint32Array(e)) : this.blocks = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0],
                this.h0 = this.h1 = this.h2 = this.h3 = this.start = this.bytes = this.hBytes = 0,
                this.finalized = this.hashed = !1,
                this.first = !0
            }
            var r = "input is invalid type", e = "object" == ("undefined" == typeof window ? "undefined" :
_typeof(window)), i = e ? window : {}, s = (i.JS_MD5_NO_WINDOW && (e = !1),
            !e && "object" == ("undefined" == typeof self ? "undefined" : _typeof(self))), h = !
i.JS_MD5_NO_NODE_JS && "object" == ("undefined" == typeof process ? "undefined" : _typeof(process))
&& process.versions && process.versions.node, f = (h ? i = commonjsGlobal : s && (i = self),
            !i.JS_MD5_NO_COMMON_JS && module.exports), o = !1, a = !i.JS_MD5_NO_ARRAY_BUFFER
&& "undefined" != typeof ArrayBuffer, n = "0123456789abcdef".split(""), u = [128, 32768, 8388608,
-2147483648], y = [0, 8, 16, 24], c = ["hex", "array", "digest", "buffer", "arrayBuffer", "base64"], p =
"ABCDEFGHIIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789+/"
.split(""), d = [], l, A, l, d, b
= (a && (A = new ArrayBuffer(68),
            l = new Uint8Array(A),
            d = new Uint32Array(A),
            !i.JS_MD5_NO_NODE_JS && Array.isArray || (Array.isArray = function(e) {
                return "[object Array]" === Object.prototype.toString.call(e)
            }
            )),
            !a || !i.JS_MD5_NO_ARRAY_BUFFER_IS_VIEW && ArrayBuffer.isView || (ArrayBuffer.isView =
function(e) {
                return "object" == _typeof(e) && e.buffer && e.buffer.constructor === ArrayBuffer
            }
            )),
            function(r) {
                return function(e) {
                    return new t(!0).update(e)[r]()
                }
            }
        ), v = function() {
            var r = b("hex");
            (r = h ? w(r) : r).create = function() {
                return new t
            }
        },
        r.update = function(e) {
            return r.create().update(e)
        }
    }
    )
    }

```

```

    }
    ;
    for (var e = 0; e < c.length; ++e) {
        var o = c[e];
        r[o] = b(o)
    }
    return r
}, w = function w(t) {
    var e = eval("require('crypto')")
    , i = eval("require('buffer').Buffer")
    , s = function(o) {
        if ("string" == typeof o)
            return e.createHash("md5").update(o, "utf8").digest("hex");
        if (null == o)
            throw r;
        return o.constructor === ArrayBuffer && (o = new Uint8Array(o)),
            ArrayBuffer.isView(o) || o.constructor === i ?
            e.createHash("md5").update(new i(o)).digest("hex") : t(o)
    };
    return s
}, _ = (t.prototype.update = function(e) {
    if (!this.finalized) {
        var t, o = _typeof(e);
        if ("string" !== o) {
            if ("object" !== o)
                throw r;
            if (null === e)
                throw r;
            if (a && e.constructor === ArrayBuffer)
                e = new Uint8Array(e);
            else if (!(Array.isArray(e) || a && ArrayBuffer.isView(e)))
                throw r;
            t = !0
        }
        for (var n, i, s = 0, l = e.length, u = this.blocks, d = this.buffer8; s < l; ) {
            if (this.hashd && (this.hashd = !1,
                u[0] = u[16],
                u[16] = u[1] = u[2] = u[3] = u[4] = u[5] = u[6] = u[7] = u[8] = u[9] = u[10] = u[11] =
                u[12] = u[13] = u[14] = u[15] = 0),
                t)
                if (a)
                    for (i = this.start; s < l && i < 64; ++s)
                        d[i++] = e[s];
                else
                    for (i = this.start; s < l && i < 64; ++s)
                        u[i >> 2] |= e[s] << y[3 & i++];
                else if (a)
                    for (i = this.start; s < l && i < 64; ++s)
                        (n = e.charCodeAtAt(s)) < 128 ? d[i++] = n : (n < 2048 ? d[i++] = 192 | n >> 6 : (n
                < 55296 || 57344 <= n ? d[i++] = 224 | n >> 12 : (n = 65536 + ((1023 & n) << 10 | 1023 & e.charCodeAtAt(+
                s)),
                    d[i++] = 240 | n >> 18,
                    d[i++] = 128 | n >> 12 & 63),
                    d[i++] = 128 | n >> 6 & 63),
                    d[i++] = 128 | 63 & n);
                else
                    for (i = this.start; s < l && i < 64; ++s)
                        (n = e.charCodeAtAt(s)) < 128 ? u[i >> 2] |= n << y[3 & i++] : (n < 2048 ? u[i >> 2]
                |= (192 | n >> 6) << y[3 & i++] : (n < 55296 || 57344 <= n ? u[i >> 2] |= (224 | n >> 12) << y[3 & i++] : (n =
                65536 + ((1023 & n) << 10 | 1023 & e.charCodeAtAt(++s)),
                    u[i >> 2] |= (240 | n >> 18) << y[3 & i++],
                    u[i >> 2] |= (128 | n >> 12 & 63) << y[3 & i++],
                    u[i >> 2] |= (128 | n >> 6 & 63) << y[3 & i++],
                    u[i >> 2] |= (128 | 63 & n) << y[3 & i++]);
            this.lastByteIndex = i,
            this.bytes += i - this.start,
            64 <= i ? (this.start = i - 64,
                this.hash(),
                this.hashd = !0) : this.start = i
        }
    }
}

```



```

    }
    return 4294967295 < this.bytes && (this.hBytes += this.bytes / 4294967296 << 0,
    this.bytes = this.bytes % 4294967296),
    this
  }
}
,
t.prototype.finalize = function() {
  var e, t;
  this.finalized || (this.finalized = !0,
  (e = this.blocks)[(t = this.lastByteIndex) >> 2] |= u[3 & t],
  56 <= t && (this.hashed || this.hash(),
  e[0] = e[16],
  e[16] = e[1] = e[2] = e[3] = e[4] = e[5] = e[6] = e[7] = e[8] = e[9] = e[10] = e[11] = e[12]
= e[13] = e[14] = e[15] = 0),
  e[14] = this.bytes << 3,
  e[15] = this.hBytes << 3 | this.bytes >>> 29,
  this.hash())
}
,
t.prototype.hash = function() {
  var e, t, r, o, n, i = this.blocks, a = this.first ? ((a = ((e = (e = i[0] - 680876937) << 7 | e >>>
25) - 271733879 << 0) ^ (t = ((t = (-271733879 ^ (r = ((r = (-1732584194 ^ 2004318071 & e) + i[1] -
117830708) << 12 | r >>> 20) + e << 0) & (-271733879 ^ e)) + i[2] - 1126478375) << 17 | t >>> 15) + r <<
0) & (r ^ e)) + i[3] - 1316259209) << 22 | a >>> 10) + t << 0 : (e = this.h0,
  a = this.h1,
  t = this.h2,
  ((a += ((e = ((e += ((r = this.h3) ^ a & (t ^ r)) + i[0] - 680876936) << 7 | e >>> 25) + a <<
0) ^ (t = ((t += (a ^ (r = ((r += (t ^ e & (a ^ t)) + i[1] - 389564586) << 12 | r >>> 20) + e << 0) & (e ^ a)) +
i[2] + 606105819) << 17 | t >>> 15) + r << 0) & (r ^ e)) + i[3] - 1044525330) << 22 | a >>> 10) + t << 0);
  a = ((a += ((e = ((e += (r ^ a & (t ^ r)) + i[4] - 176418897) << 7 | e >>> 25) + a << 0) ^ (t
= ((t += (a ^ (r = ((r += (t ^ e & (a ^ t)) + i[5] + 1200080426) << 12 | r >>> 20) + e << 0) & (e ^ a)) + i[6] -
1473231341) << 17 | t >>> 15) + r << 0) & (r ^ e)) + i[7] - 45705983) << 22 | a >>> 10) + t << 0,
  a = ((a += ((e = ((e += (r ^ a & (t ^ r)) + i[8] + 1770035416) << 7 | e >>> 25) + a << 0) ^ (t
= ((t += (a ^ (r = ((r += (t ^ e & (a ^ t)) + i[9] - 1958414417) << 12 | r >>> 20) + e << 0) & (e ^ a)) + i[10]
- 42063) << 17 | t >>> 15) + r << 0) & (r ^ e)) + i[11] - 1990404162) << 22 | a >>> 10) + t << 0,
  a = ((a += ((e = ((e += (r ^ a & (t ^ r)) + i[12] + 1804603682) << 7 | e >>> 25) + a << 0) ^
(t = ((t += (a ^ (r = ((r += (t ^ e & (a ^ t)) + i[13] - 40341101) << 12 | r >>> 20) + e << 0) & (e ^ a)) +
i[14] - 1502002290) << 17 | t >>> 15) + r << 0) & (r ^ e)) + i[15] + 1236535329) << 22 | a >>> 10) + t << 0,
  a = ((a += ((r = ((r += (a ^ t & ((e = ((e += (t ^ r & (a ^ t)) + i[1] - 165796510) << 5 | e
>>> 27) + a << 0) ^ a)) + i[6] - 1069501632) << 9 | r >>> 23) + e << 0) ^ e & ((t = ((t += (e ^ a & (r ^ e)) +
i[11] + 643717713) << 14 | t >>> 18) + r << 0) ^ r)) + i[0] - 373897302) << 20 | a >>> 12) + t << 0,
  a = ((a += ((r = ((r += (a ^ t & ((e = ((e += (t ^ r & (a ^ t)) + i[5] - 701558691) << 5 | e
>>> 27) + a << 0) ^ a)) + i[10] + 38016083) << 9 | r >>> 23) + e << 0) ^ e & ((t = ((t += (e ^ a & (r ^ e)) +
i[15] - 660478335) << 14 | t >>> 18) + r << 0) ^ r)) + i[4] - 405537848) << 20 | a >>> 12) + t << 0,
  a = ((a += ((r = ((r += (a ^ t & ((e = ((e += (t ^ r & (a ^ t)) + i[9] + 568446438) << 5 | e
>>> 27) + a << 0) ^ a)) + i[14] - 1019803690) << 9 | r >>> 23) + e << 0) ^ e & ((t = ((t += (e ^ a & (r ^ e))
+ i[3] - 187363961) << 14 | t >>> 18) + r << 0) ^ r)) + i[8] + 1163531501) << 20 | a >>> 12) + t << 0,
  a = ((a += ((r = ((r += (a ^ t & ((e = ((e += (t ^ r & (a ^ t)) + i[13] - 1444681467) << 5 | e
>>> 27) + a << 0) ^ a)) + i[2] - 51403784) << 9 | r >>> 23) + e << 0) ^ e & ((t = ((t += (e ^ a & (r ^ e)) +
i[7] + 1735328473) << 14 | t >>> 18) + r << 0) ^ r)) + i[12] - 1926607734) << 20 | a >>> 12) + t << 0,
  a = ((a += ((n = (r = ((r += ((o = a ^ t) ^ (e = ((e += (o ^ r) + i[5] - 378558) << 4 | e >>>
28) + a << 0)) + i[8] - 2022574463) << 11 | r >>> 21) + e << 0) ^ e) ^ (t = ((t += (n ^ a) + i[11] +
1839030562) << 16 | t >>> 16) + r << 0)) + i[14] - 35309556) << 23 | a >>> 9) + t << 0,
  a = ((a += ((n = (r = ((r += ((o = a ^ t) ^ (e = ((e += (o ^ r) + i[1] - 1530992060) << 4 | e
>>> 28) + a << 0)) + i[4] + 1272893353) << 11 | r >>> 21) + e << 0) ^ e) ^ (t = ((t += (n ^ a) + i[7] -
155497632) << 16 | t >>> 16) + r << 0)) + i[10] - 1094730640) << 23 | a >>> 9) + t << 0,
  a = ((a += ((n = (r = ((r += ((o = a ^ t) ^ (e = ((e += (o ^ r) + i[13] + 681279174) << 4 | e
>>> 28) + a << 0)) + i[0] - 358537222) << 11 | r >>> 21) + e << 0) ^ e) ^ (t = ((t += (n ^ a) + i[3] -
722521979) << 16 | t >>> 16) + r << 0)) + i[6] + 76029189) << 23 | a >>> 9) + t << 0,
  a = ((a += ((n = (r = ((r += ((o = a ^ t) ^ (e = ((e += (o ^ r) + i[9] - 640364487) << 4 | e
>>> 28) + a << 0)) + i[12] - 421815835) << 11 | r >>> 21) + e << 0) ^ e) ^ (t = ((t += (n ^ a) + i[15] +
530742520) << 16 | t >>> 16) + r << 0)) + i[2] - 995338651) << 23 | a >>> 9) + t << 0,
  a = ((a += ((r = ((r += (a ^ (e = ((e += (t ^ (a | ~r)) + i[0] - 198630844) << 6 | e >>> 26) +
a << 0) | ~t)) + i[7] + 1126891415) << 10 | r >>> 22) + e << 0) ^ ((t = ((t += (e ^ (r | ~a)) + i[14] -
1416354905) << 15 | t >>> 17) + r << 0) | ~e)) + i[5] - 57434055) << 21 | a >>> 11) + t << 0,
  a = ((a += ((r = ((r += (a ^ (e = ((e += (t ^ (a | ~r)) + i[12] + 1700485571) << 6 | e >>>
26) + a << 0) | ~t)) + i[3] - 1894986606) << 10 | r >>> 22) + e << 0) ^ ((t = ((t += (e ^ (r | ~a)) + i[10] -
1051523) << 15 | t >>> 17) + r << 0) | ~e)) + i[1] - 2054922799) << 21 | a >>> 11) + t << 0,

```

```

        a = ((a += ((r = ((r += (a ^ ((e = ((e += (t ^ (a | ~r)) + i[8] + 1873313359) << 6 | e >>> 26)
+ a << 0) | ~t)) + i[15] - 30611744) << 10 | r >>> 22) + e << 0) ^ ((t += (e ^ (r | ~a)) + i[6] -
1560198380) << 15 | t >>> 17) + r << 0) | ~e) + i[13] + 1309151649) << 21 | a >>> 11) + t << 0,
        a = ((a += ((r = ((r += (a ^ ((e = ((e += (t ^ (a | ~r)) + i[4] - 145523070) << 6 | e >>> 26) +
a << 0) | ~t)) + i[11] - 1120210379) << 10 | r >>> 22) + e << 0) ^ ((t += (e ^ (r | ~a)) + i[2] +
718787259) << 15 | t >>> 17) + r << 0) | ~e) + i[9] - 343485551) << 21 | a >>> 11) + t << 0,
        this.first ? (this.h0 = e + 1732584193 << 0,
        this.h1 = a - 271733879 << 0,
        this.h2 = t - 1732584194 << 0,
        this.h3 = r + 271733878 << 0,
        this.first = !1) : (this.h0 = this.h0 + e << 0,
        this.h1 = this.h1 + a << 0,
        this.h2 = this.h2 + t << 0,
        this.h3 = this.h3 + r << 0)
    }
    ,
    t.prototype.hex = function() {
        this.finalize();
        var e = this.h0
        , t = this.h1
        , r = this.h2
        , o = this.h3;
        return n[e >> 4 & 15] + n[15 & e] + n[e >> 12 & 15] + n[e >> 8 & 15] + n[e >> 20 & 15] +
n[e >> 16 & 15] + n[e >> 28 & 15] + n[e >> 24 & 15] + n[t >> 4 & 15] + n[15 & t] + n[t >> 12 & 15] + n[t
>> 8 & 15] + n[t >> 20 & 15] + n[t >> 16 & 15] + n[t >> 28 & 15] + n[t >> 24 & 15] + n[r >> 4 & 15] + n[15
& r] + n[r >> 12 & 15] + n[r >> 8 & 15] + n[r >> 20 & 15] + n[r >> 16 & 15] + n[r >> 28 & 15] + n[r >> 24
& 15] + n[o >> 4 & 15] + n[15 & o] + n[o >> 12 & 15] + n[o >> 8 & 15] + n[o >> 20 & 15] + n[o >> 16 &
15] + n[o >> 28 & 15] + n[o >> 24 & 15]
    }
    ,
    t.prototype.toString = t.prototype.hex,
    t.prototype.digest = function() {
        this.finalize();
        var e = this.h0
        , t = this.h1
        , r = this.h2
        , o = this.h3;
        return [255 & e, e >> 8 & 255, e >> 16 & 255, e >> 24 & 255, 255 & t, t >> 8 & 255, t >> 16
& 255, t >> 24 & 255, 255 & r, r >> 8 & 255, r >> 16 & 255, r >> 24 & 255, 255 & o, o >> 8 & 255, o >> 16 &
255, o >> 24 & 255]
    }
    ,
    t.prototype.array = t.prototype.digest,
    t.prototype.arrayBuffer = function() {
        this.finalize();
        var e = new ArrayBuffer(16)
        , t = new Uint32Array(e);
        return t[0] = this.h0,
        t[1] = this.h1,
        t[2] = this.h2,
        t[3] = this.h3,
        e
    }
    ,
    t.prototype.buffer = t.prototype.arrayBuffer,
    t.prototype.base64 = function() {
        for (var e, t, r, o = "", n = this.array(), i = 0; i < 15; )
            e = n[i++],
            t = n[i++],
            r = n[i++],
            o += p[e >>> 2] + p[63 & (e << 4 | t >>> 4)] + p[63 & (t << 2 | r >>> 6)] + p[63 & r];
        return e = n[i],
        o + (p[e >>> 2] + p[e << 4 & 63] + "==")
    }
    ,
    v());
    f ? module.exports = _ : i.md5 = _
}()
});

```

```

function _typeof(e) {
  return (_typeof = "function" == typeof Symbol && "symbol" == typeof Symbol.iterator ?
function(e) {
  return typeof e
}
: function(e) {
  return e && "function" == typeof Symbol && e.constructor === Symbol && e !==
Symbol.prototype ? "symbol" : typeof e
}
)(e)
}

function createCommonjsModule(e, t) {
  return e(t = {
    exports: {}
  }, t.exports),
  t.exports
}

function Str2Bytes(e) {
  var t = 0
  , r = e.length;
  if (r % 2 != 0)
    return null;
  r /= 2;
  for (var o = new Array, n = 0; n < r; n++) {
    var i = e.substr(t, 2)
    , i = parseInt(i, 16);
    o.push(i),
    t += 2
  }
  return o
}

</script>
</head>

<body>
<p>projectId: <input type="input" id="projectId"></input></p>
<p>token: <input type="input" id="token"></input></p>
<p><input type="file" id="file"></input></p>
<button onclick="startUpload()">上传</button>
</body>
</html>

```

19.7.4 Python

分段上传Python语言的示例代码，如下所示：

```

import base64
import hashlib
import os
import re

import xml.etree.ElementTree as ET
import requests

from huaweicloudsdkcore.auth.credentials import BasicCredentials
from huaweicloudsdkvod.v1 import *
from huaweicloudsdkvod.v1.region.vod_region import VodRegion

class PartUploadDemo:
    """
    分段上传示例
    """
    # 设置缓冲区大小，每次读取文件分段的大小，根据情况自行设定
    # 1MB

```

```

buffer_size = 1024 * 1024

# 区域
region_north4 = "cn-north-4"
region_north1 = "cn-north-1"
region_east2 = "cn-east-2"

region = ""

# ak/sk, 示例以ak/sk做认证鉴权
ak = ""
sk = ""

def __init__(self):
    pass

def upload_file(self, file_path):
    """
    分段上传
    :param file_path: 文件的本地路径
    :type file_path: str
    :return:
    """
    # 校验一下文件路径和文件
    if not self.valid_file(file_path):
        return

    # 取文件名
    filename = os.path.basename(file_path)

    # 此处仅以MP4文件示例, 其他格式可参考官网说明
    video_type = "MP4"
    file_content_type = "video/mp4"

    print("开始上传媒资: " + filename)
    # 1. 初始化鉴权并获取vodClient
    client = self.create_vod_client()

    # 2. 创建点播媒资
    asset_response = self.create_asset(client=client,
                                       file_name=filename,
                                       video_type=video_type)

    # 3. 获取初始化上传任务授权
    init_auth_response = self.init_part_upload_authority(client=client,
                                                         asset_response=asset_response,
                                                         file_content_type=file_content_type)

    # 4. 初始化上传任务
    upload_id = self.init_part_upload(sign_str=init_auth_response.sign_str,
                                       file_content_type=file_content_type)

    # 文件分段计数
    part_number = 1

    # 7. 读取文件内容, 循环5-6步上传所有分段
    with open(file_path, 'rb') as f:
        for chunk in iter(lambda: f.read(self.buffer_size), b''):
            # 先md5,再base64,生成content_md5
            md5 = hashlib.md5()
            md5.update(chunk)
            content_md5 = str(base64.b64encode(md5.digest()), 'utf-8')
            # print(content_md5)

            # 5. 获取上传分段的授权
            upload_auth_response = self.get_part_upload_authority(client=client,
                                                                  asset_response=asset_response,
                                                                  file_content_type=file_content_type,
                                                                  content_md5=content_md5,

```

```

        upload_id=upload_id,
        part_number=part_number)

# 6.上传分段
self.upload_part_file(sign_str=upload_auth_response.sign_str,
                      chunk=chunk,
                      content_md5=content_md5,
                      part_number=part_number)

# 段号自增
part_number += 1

# 8.获取已上传分段的授权
list_part_upload_authority_response = self.list_uploaded_part_authority(client=client,
                               asset_response=asset_response,
                               upload_id=upload_id)

# 9.获取已上传的分段
part_info = self.list_uploaded_part(sign_str=list_part_upload_authority_response.sign_str)

# 10.获取合并段授权
merge_part_upload_authority_response = self.merge_uploaded_part_authority(client=client,
                                  asset_response=asset_response,
                                  upload_id=upload_id)

# 11. 合并上传分段
self.merge_uploaded_part(sign_str=merge_part_upload_authority_response.sign_str,
                        part_info=part_info)

# 12. 确认媒资上传
self.confirm_uploaded(client=client, asset_response=asset_response)
print("上传媒资结束 assetId:" + asset_response.asset_id)

# 校验文件是否存在
def valid_file(self, file_path):
    valid_result = True
    if not file_path:
        print("路径为空")
        valid_result = False
    elif os.path.isdir(file_path):
        print("是个目录")
        valid_result = False
    elif not os.path.isfile(file_path):
        print("文件不存在")
        valid_result = False
    return valid_result

# 1.初始化鉴权
def create_vod_client(self):
    print("初始化鉴权。。。")
    credentials = BasicCredentials(self.ak, self.sk)
    client = VodClient.new_builder() \
        .with_credentials(credentials) \
        .with_region(VodRegion.value_of(self.region)) \
        .build()
    return client

def create_asset(self, client, file_name, video_type):
    """
    2.创建点播媒资
    :param client
    :param file_name 音视频文件名
    :type file_name: str
    :param video_type 上传音视频文件的格式
    :type video_type: str
    """
    print("create_asset start; ")
    create_asset_request = CreateAssetByFileUploadRequest()
    # 设置创建媒资参数, 创建媒资参数最小集, 其他参数参考官网文档按需添加

```

```

create_asset_request.body = CreateAssetByFileUploadReq(
    video_type=video_type,
    video_name=file_name,
    title=file_name
)
# 调用创建媒资方法
asset_response = client.create_asset_by_file_upload(create_asset_request)
print("create_asset end")
return asset_response

def init_part_upload_authority(self, client, asset_response, file_content_type):
    """
    3.获取初始化上传任务授权
    :param client:
    :param asset_response: 创建媒资返回的结果
    :param file_content_type: 文件类型对应的content-type,如MP4对应video/mp4
    :type file_content_type: str
    :return:
    """
    print("获取初始化上传任务授权 init_part_upload_authority start")
    init_auth_request = ShowAssetTempAuthorityRequest()
    # 设置初始化参数
    init_auth_request.http_verb = "POST"
    init_auth_request.bucket = asset_response.target.bucket
    init_auth_request.object_key = asset_response.target.object
    init_auth_request.content_type = file_content_type
    # 发送初始化请求
    init_auth_response = client.show_asset_temp_authority(init_auth_request)
    print("获取初始化上传任务授权 init_part_upload_authority end")
    return init_auth_response

def init_part_upload(self, sign_str, file_content_type):
    """
    4.初始化上传任务
    :param sign_str: 第3步返回结果中的sign_str, 初始化上传任务连接
    :type sign_str: str
    :param file_content_type: 文件类型对应的content-type,如MP4对应video/mp4
    :type file_content_type: str
    :return: 返回upload_id
    """
    print("初始化分段上传 init_part_upload start")
    # 发送初始化请求
    init_response = requests.request(method="POST",
                                    url=sign_str,
                                    headers={"Content-Type": file_content_type})
    print(init_response.text)
    # 解析响应内容, 获取uploadId
    root = ET.fromstring(init_response.text)
    namespace_str = root.tag
    match = re.search(r'\{(.*)\}', namespace_str)
    namespace_uri = match.group(1)
    upload_id = root.find("{ " + namespace_uri + "}UploadId").text
    print("初始化分段上传 init_part_upload end; UploadId:" + upload_id)
    return upload_id

def get_part_upload_authority(self, client, asset_response, file_content_type, content_md5, upload_id,
part_number):
    """
    5.获取上传分段的授权
    :param client:
    :param asset_response: 创建媒资返回的结果
    :param file_content_type: 文件类型对应的content-type,如MP4对应video/mp4
    :type file_content_type: str
    :param content_md5: 文件当前分段的content-md5值
    :type content_md5: str
    :param upload_id:
    :type upload_id: str
    :param part_number: 段号
    :type part_number: int
    """

```

```

:return:
"""
print("获取分段上传授权 get_part_upload_authority start; partNumber:", part_number)
upload_auth_request = ShowAssetTempAuthorityRequest()
# 设置上传授权参数
upload_auth_request.http_verb = "PUT"
upload_auth_request.bucket = asset_response.target.bucket
upload_auth_request.object_key = asset_response.target.object
upload_auth_request.content_type = file_content_type
upload_auth_request.content_md5 = content_md5
upload_auth_request.upload_id = upload_id
upload_auth_request.part_number = part_number
upload_auth_response = client.show_asset_temp_authority(upload_auth_request)
print(upload_auth_response)
print("获取分段上传授权 get_part_upload_authority end; partNumber:", part_number)
return upload_auth_response

def upload_part_file(self, sign_str, chunk, content_md5, part_number):
    """
    6.上传分段
    :param sign_str: 第5步返回结果中的sign_str, 上传连接
    :type sign_str: str
    :param chunk: 文件当前分段的二进制数据
    :type chunk: bytes
    :param content_md5: 文件当前分段的content-md5
    :type content_md5: str
    :param part_number: 当前段号
    :type part_number: int
    :return:
    """
    print("上传分段 upload_part_file start; partNumber:", part_number)
    # 发送上传分段请求
    upload_response = requests.request(method="PUT",
                                      url=sign_str,
                                      headers={
                                          "Content-Type": "application/octet-stream",
                                          "Content-MD5": content_md5
                                      },
                                      data=chunk)
    if upload_response.status_code != 200:
        print("上传分段 end;上传失败! partNumber:", part_number)
        raise Exception("上传分段 end;上传失败! partNumber:", part_number)
    print("上传分段 upload_part_file end! partNumber:", part_number)

def list_uploaded_part_authority(self, client, asset_response, upload_id):
    """
    8.获取已上传分段的授权
    :param client:
    :param asset_response: 创建媒资返回的结果
    :param upload_id:
    :return:
    """
    print("获取列举已上传段的授权 list_uploaded_part_authority start")
    # 设置参数
    list_upload_part_auth_request = ShowAssetTempAuthorityRequest()
    list_upload_part_auth_request.http_verb = "GET"
    list_upload_part_auth_request.bucket = asset_response.target.bucket
    list_upload_part_auth_request.object_key = asset_response.target.object
    list_upload_part_auth_request.upload_id = upload_id
    list_upload_part_auth_response = client.show_asset_temp_authority(list_upload_part_auth_request)
    print(list_upload_part_auth_response)
    print("获取列举已上传段的授权 list_uploaded_part_authority end; ")
    return list_upload_part_auth_response

def list_uploaded_part(self, sign_str):
    """
    9.获取已上传的分段
    :param sign_str: 第8步返回的授权连接
    :type sign_str: str
    """

```

```

:return:
"""
print("查询已上传的分段 list_uploaded_part start")
# 查询分段的起始段号
part_number_marker = 0
# 组装合并段时根节点
merger_root = ET.Element("CompleteMultipartUpload")
# 每次取列表最大返回1000段信息，分段超过1000要多次调用列举分段接口
while True:
    # 列举分段
    list_upload_part_auth_response = requests.request(method="GET",
                                                    url=sign_str + "&part-number-marker=" + str(
                                                        part_number_marker))
    print(list_upload_part_auth_response)

    # 按照xml格式化响应内容
    response_document = ET.fromstring(list_upload_part_auth_response.text)
    # 解析xml内容，取xmlns信息
    namespace_str_m = response_document.tag
    match_m = re.search(r'\{(.*)}', namespace_str_m)
    namespace_uri_m = match_m.group(1)

    # 遍历Part节点
    for part in response_document.findall("{ " + namespace_uri_m + "}Part"):
        # 取PartNumber和ETag信息
        part_number_value = part.find("{ " + namespace_uri_m + "}PartNumber").text
        e_tag_value = part.find("{ " + namespace_uri_m + "}ETag").text

        # 组装合并信息
        # 在根节点下创建Part节点
        part_node = ET.SubElement(merger_root, "Part")
        # Part节点下创建PartNumber，并设置PartNumber值
        part_number_node = ET.SubElement(part_node, "PartNumber")
        part_number_node.text = part_number_value
        # Part节点下创建ETag，并设置ETag值
        e_tag_node = ET.SubElement(part_node, "ETag")
        e_tag_node.text = e_tag_value

        # 查找设置下个列表起始值
        part_number_marker_element = response_document.find("{ " + namespace_uri_m +
    }NextPartNumberMarker")
        part_number_marker = int(part_number_marker_element.text)
        # 当段号不为1000整数倍，则已取完分段
        if part_number_marker % 1000 != 0:
            break

    part_info = ET.tostring(merger_root, encoding='utf8')
    print(part_info)
    print("查询已上传的分段 list_uploaded_part end")
    return part_info

def merge_uploaded_part_authority(self, client, asset_response, upload_id):
    """
    10.获取合并段授权
    :param client:
    :param asset_response: 创建媒资返回的结果
    :param upload_id: upload_id
    :type upload_id: str
    :return:
    """
    print("获取合并段授权 merge_uploaded_part_authority start")
    # 设置参数
    merger_part_upload_auth_request = ShowAssetTempAuthorityRequest()
    merger_part_upload_auth_request.http_verb = "POST"
    merger_part_upload_auth_request.bucket = asset_response.target.bucket
    merger_part_upload_auth_request.object_key = asset_response.target.object
    merger_part_upload_auth_request.upload_id = upload_id
    merger_part_upload_auth_response =
client.show_asset_temp_authority(merger_part_upload_auth_request)

```



```

print(merger_part_upload_auth_response)
print("获取合并段授权 merge_uploaded_part_authority end")
return merger_part_upload_auth_response

def merge_uploaded_part(self, sign_str, part_info):
    """
    11.合并上传分段
    :param sign_str: 第10步返回的合并授权连接
    :type sign_str: str
    :param part_info: 合并段信息
    :type part_info: str
    :return:
    """
    print("合并分段 start")
    # 请求消息头中增加 "Content-Type" , 值设置为 "application/xml" 。
    merger_part_upload_response = requests.request(method="POST",
                                                    url=sign_str,
                                                    headers={"Content-Type": "application/xml"},
                                                    data=part_info)

    print(merger_part_upload_response)
    if merger_part_upload_response.status_code != 200:
        print("合并分段end;合并分段失败")
    print("合并分段 end")

# 12.确认媒资上传
def confirm_uploaded(self, client, asset_response):
    print("确认上传完成start")
    confirm_asset_upload_request = ConfirmAssetUploadRequest()
    confirm_asset_upload_request.body = ConfirmAssetUploadReq(status="CREATED",
                                                              asset_id=asset_response.asset_id)

    confirm_asset_upload_response = client.confirm_asset_upload(confirm_asset_upload_request)
    print(confirm_asset_upload_response)

if __name__ == '__main__':
    # 本地要上传的媒资路径
    filePath = ""
    partUploadDemo = PartUploadDemo()
    partUploadDemo.ak = ""
    partUploadDemo.sk = ""
    partUploadDemo.region = partUploadDemo.region_north4
    # 上传媒资文件
    partUploadDemo.upload_file(file_path=filePath)
    
```

19.7.5 Go

分段上传Go语言的示例代码，如下所示：

```

package main

import (
    "bytes"
    "crypto/md5"
    "encoding/base64"
    "encoding/xml"
    "errors"
    "fmt"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/core/auth/basic"
    vod "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/vod/v1"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/vod/v1/model"
    "github.com/huaweicloud/huaweicloud-sdk-go-v3/services/vod/v1/region"
    "io"
    "net/http"
    "os"
)

// 区域
const regionNorth4 = "cn-north-4"
const regionNorth1 = "cn-north-1"
    
```

```

const regionEast2 = "cn-east-2"

// ak/sk
const ak = ""
const sk = ""

// 设置缓冲区大小, 每次读取文件分段的大小, 根据情况自行设定
// 1M
const bufferSize = 1024 * 1024

/*
分段上传示例
*/
func main() {
    // 本地要上传的媒资路径
    filePath := ""

    // 上传媒资文件
    partUpload(filePath)
}

/*
分段上传
filePath 上传文件的本地路径
*/
func partUpload(filePath string) {
    // 校验文件路径和文件
    fileInfo := validFile(filePath)
    fileName := fileInfo.Name()
    fmt.Println(fileName)

    file, err := os.Open(filePath)
    if err != nil {
        fmt.Println("Error:", err)
        return
    }
    defer func(file *os.File) {
        err := file.Close()
        if err != nil {
            panic(err)
        }
    }(file)

    // 此处仅以MP4文件示例, 其他格式可参考官网说明
    fileType := "MP4"
    fileContentType := "video/mp4"

    // 1.初始化鉴权并获取vodClient
    client := createVodClient()
    // 2.创建点播媒资
    fmt.Println("开始创建媒资:" + fileName)
    assetResponse := createAsset(client, fileName, fileName, fileType)

    // 3.获取初始化上传任务授权
    initAuthResponse := initPartUploadAuthority(client, assetResponse, fileContentType)

    // 4.初始化上传任务
    uploadId := initPartUpload(initAuthResponse.SignStr, fileContentType)

    // 文件分段计数
    partNumber := 1

    // 7.读取文件内容, 循环5-6步上传所有分段
    for {
        buf := make([]byte, bufferSize)
        n, err := file.Read(buf)
        if n == 0 || err != nil || err == io.EOF {
            break
        }
    }
}

```

```

// 先md5, 再base64 encode
h := md5.New()
h.Write(buf[:n])
data := h.Sum(nil)
contentMd5 := base64.StdEncoding.EncodeToString(data)
fmt.Println("该文件第", partNumber, "段MD5为:", contentMd5)

// 5.获取上传分段的授权
partUploadAuthorityResponse := getPartUploadAuthority(client, fileContentType, assetResponse,
    contentMd5, uploadId, partNumber)

// 6.上传分段
uploadPartFile(partUploadAuthorityResponse.SignStr, buf[:n], contentMd5)

// 段号自增
partNumber++
}

// 8.获取已上传分段的授权
listPartUploadAuthorityResponse := listUploadedPartAuthority(client, assetResponse, uploadId)

// 9.获取已上传的分段
partInfo := listUploadedPart(listPartUploadAuthorityResponse.SignStr)

// 10.获取合并段授权
mergePartUploadAuthorityResponse := mergeUploadedPartAuthority(client, assetResponse, uploadId)

// 11.合并上传分段
mergeUploadedPart(mergePartUploadAuthorityResponse.SignStr, partInfo)

// 12.确认媒资上传
confirmUploaded(client, assetResponse)

fmt.Println("创建媒资结束 assetId:", *assetResponse.AssetID)
}

/*
校验文件路径和文件
filePath: 文件路径
*/
func validFile(filePath string) os.FileInfo {
    // 检查文件是否存在
    fileInfo, err := os.Stat(filePath)
    if os.IsNotExist(err) {
        fmt.Println("文件不存在")
        // 抛出错误
        panic(errors.New("文件不存在"))
    }
    if err != nil {
        fmt.Println("发生错误:", err)
        panic(err)
    }
    return fileInfo
}

/*
1.构建鉴权
*/
func createVodClient() *vod.VodClient {
    auth, _ := basic.NewCredentialsBuilder().
        WithAk(ak).
        WithSk(sk).
        SafeBuild()
    reg, _ := region.SafeValueOf(regionNorth4)

    build, _ := vod.VodClientBuilder().
        WithRegion(reg).
        WithCredential(auth).
        SafeBuild()
}

```

```

    client := vod.NewVodClient(build)
    return client
}

/*
2.创建媒资
videoName: 名称
title: 标题
videoType: 文件类型
*/
func createAsset(client *vod.VodClient, videoName string, title string,
    videoType string) *model.CreateAssetByFileUploadResponse {
    request := &model.CreateAssetByFileUploadRequest{}
    request.Body = &model.CreateAssetByFileUploadReq{
        VideoName: videoName,
        Title: title,
        VideoType: videoType,
    }
    response, err := client.CreateAssetByFileUpload(request)
    if err == nil {
        fmt.Println(response)
    } else {
        fmt.Println(err)
        panic(err)
    }
    return response
}

/*
3.获取初始化上传任务授权
client
assetResponse
*/
func initPartUploadAuthority(client *vod.VodClient, assetResponse *model.CreateAssetByFileUploadResponse,
    fileContentType string) *model.ShowAssetTempAuthorityResponse {
    fmt.Println("获取初始化上传任务授权 initPartUploadAuthority start")
    // 设置参数
    request := &model.ShowAssetTempAuthorityRequest{}
    request.HttpVerb = "POST"
    request.Bucket = assetResponse.Target.Bucket
    request.ObjectKey = assetResponse.Target.Object
    request.ContentType = &fileContentType
    // 发送初始化请求
    response, err := client.ShowAssetTempAuthority(request)
    if err == nil {
        fmt.Print(response)
    } else {
        fmt.Println(err)
        panic(err)
    }
    fmt.Println("获取初始化上传任务授权 initPartUploadAuthority end\n")
    return response
}

/*
4.初始化分段上传
signStr 第3步返回的带签名的初始化url
contentType 媒体文件格式
return uploadId
*/
func initPartUpload(signStr *string, contentType string) string {
    fmt.Println("初始化分段上传 initPartUpload start")
    // 创建一个新的HTTP客户端
    client := &http.Client{}
    // 创建一个新的请求
    req, err := http.NewRequest("POST", *signStr, nil)
    req.Header.Set("Content-Type", contentType)
}

```

```

// 发送请求
resp, err := client.Do(req)
if err != nil {
    fmt.Println("Error sending request:", err)
    panic(err)
}
defer func(Body io.ReadCloser) {
    err := Body.Close()
    if err != nil {
        panic(err)
    }
}(resp.Body)

// 读取响应体
body, _ := io.ReadAll(resp.Body)
// 打印响应体
fmt.Println(string(body))

// 解析响应体, 返回结果为xml文本, 按格式解析
var initiateMultipartUploadResult InitiateMultipartUploadResult
err = xml.Unmarshal(body, &initiateMultipartUploadResult)
if err != nil {
    panic(err)
}
fmt.Printf("Bucket:%s, Key:%s, UploadId:%s\n", initiateMultipartUploadResult.Bucket,
    initiateMultipartUploadResult.Key, initiateMultipartUploadResult.UploadId)
fmt.Println("初始化分段上传 initPartUpload end\n")
return initiateMultipartUploadResult.UploadId
}

/*
5.获取分段上传授权
client
fileContentType 媒体文件格式
assetResponse 创建媒资返回的结果
contentMd5 当前分段计算出的contentMd5
uploadId
partNumber 段号
return 授权结果
*/
func getPartUploadAuthority(client *vod.VodClient, fileContentType string, assetResponse
*model.CreateAssetByFileUploadResponse,
contentMd5 string, uploadId string, partNumber int) *model.ShowAssetTempAuthorityResponse {
    fmt.Println("获取分段上传授权 getPartUploadAuthority start; partNumber:", partNumber)
    // 设置参数
    request := &model.ShowAssetTempAuthorityRequest{}
    request.HttpVerb = "PUT"
    request.Bucket = assetResponse.Target.Bucket
    request.ObjectKey = assetResponse.Target.Object
    request.ContentType = &fileContentType
    request.ContentMd5 = &contentMd5
    request.UploadId = &uploadId
    partNumberRequest := int32(partNumber)
    request.PartNumber = &partNumberRequest
    // 发送请求
    response, err := client.ShowAssetTempAuthority(request)
    if err == nil {
        fmt.Print(response)
    } else {
        fmt.Println(err)
        panic(err)
    }
    fmt.Println("获取分段上传授权 getPartUploadAuthority end; partNumber:", partNumber, "\n")
    return response
}

/*
6.上传分段
signStr 第5步返回的带签名的上传url

```

```

fileByte 当前分段数据
contentMd5 当前分段contentMd5
*/
func uploadPartFile(signStr *string, fileByte []byte, contentMd5 string) {
    fmt.Println("上传分段 uploadPartFile start")

    // 创建一个新的HTTP客户端
    client := &http.Client{}
    // 创建一个新的请求
    req, err := http.NewRequest("PUT", *signStr, bytes.NewBuffer(fileByte))
    req.Header.Set("Content-MD5", contentMd5)
    req.Header.Set("Content-Type", "application/octet-stream")

    // 发送请求
    resp, err := client.Do(req)
    if err != nil {
        fmt.Println("Error sending request:", err)
        panic(err)
    }
    defer func(Body io.ReadCloser) {
        err := Body.Close()
        if err != nil {
            panic(err)
        }
    }(resp.Body)
    // 读取响应体
    body, _ := io.ReadAll(resp.Body)
    // 打印响应体
    fmt.Println(string(body))
    if resp.StatusCode != 200 {
        panic("上传文件失败! ")
    }
    fmt.Println("上传分段 uploadPartFile end \n")
}

/*
8.获取列举已上传段的授权
client
assetResponse 创建媒资响应结果
uploadId
return
*/
func listUploadedPartAuthority(client *vod.VodClient, assetResponse
*model.CreateAssetByFileUploadResponse,
uploadId string) *model.ShowAssetTempAuthorityResponse {
    fmt.Println("获取列举已上传段的授权 listUploadedPartAuthority start")
    // 设置参数
    request := &model.ShowAssetTempAuthorityRequest{}
    request.HttpVerb = "GET"
    request.Bucket = assetResponse.Target.Bucket
    request.ObjectKey = assetResponse.Target.Object
    request.UploadId = &uploadId
    // 发送请求
    response, err := client.ShowAssetTempAuthority(request)
    if err == nil {
        fmt.Print(response)
    } else {
        fmt.Println(err)
        panic(err)
    }
    fmt.Println("获取列举已上传段的授权 listUploadedPartAuthority end\n")
    return response
}

/*
9.查询已上传的分段
signStr 第8步返回的带签名的查询url
return
*/

```

```

func listUploadedPart(signStr *string) string {
    fmt.Println("查询已上传的分段 listUploadedPart start")
    // 查询分段的起始段号
    partNumberMarker := 0
    // 组装合并段参数, xml格式
    result := "<CompleteMultipartUpload>"
    // 创建一个新的HTTP客户端
    client := &http.Client{}
    for {
        // 创建一个新的请求
        url := *signStr + "&part-number-marker=" + fmt.Sprintf("%d", partNumberMarker)
        req, _ := http.NewRequest("GET", url, nil)
        // 发送请求
        resp, _ := client.Do(req)
        // 读取响应体
        body, _ := io.ReadAll(resp.Body)
        // 打印响应体
        fmt.Println(string(body))
        // 解析响应结果, 转换xml格式文本
        var listPartsResult ListPartsResult
        _ = xml.Unmarshal(body, &listPartsResult)
        // 响应结果中没有parts跳出
        if len(listPartsResult.Parts) < 1 {
            break
        }
        // 循环parts组装Part数据
        for _, part := range listPartsResult.Parts {
            num := part.PartNumber
            tag := part.Etag
            result += "<Part>" +
                "<PartNumber>" +
                fmt.Sprintf("%d", num) +
                "</PartNumber>" +
                "<ETag>" +
                tag +
                "</ETag>" +
                "</Part>"
        }
        // 响应中下个段号值设为起始段号并再次请求
        partNumberMarker = listPartsResult.NextPartNumberMarker
        if partNumberMarker%1000 != 0 {
            break
        }
    }
    result += "</CompleteMultipartUpload>"
    fmt.Println(result)
    fmt.Println("查询已上传的分段 listUploadedPart end\n")
    return result
}

/*
10.获取合并段授权
client
assetResponse
uploadId
return
*/
func mergeUploadedPartAuthority(client *vod.VodClient, assetResponse
*model.CreateAssetByFileUploadResponse,
uploadId string) *model.ShowAssetTempAuthorityResponse {
    fmt.Println("获取合并段授权 mergeUploadedPartAuthority start")
    // 设置参数
    request := &model.ShowAssetTempAuthorityRequest{}
    request.HttpVerb = "POST"
    request.Bucket = assetResponse.Target.Bucket
    request.ObjectKey = assetResponse.Target.Object
    request.UploadId = &uploadId
    // 发送请求
    response, err := client.ShowAssetTempAuthority(request)
}

```

```

    if err == nil {
        fmt.Print(response)
    } else {
        fmt.Println(err)
        panic(err)
    }
    fmt.Println("获取合并段授权 mergeUploadedPartAuthority end\n")
    return response
}

/*
11.合并分段
signStr 第10步返回的带签名的合并url
partInfo 需要合并段的数据
*/
func mergeUploadedPart(signStr *string, partInfo string) {
    fmt.Println("合并分段 mergeUploadedPart start")
    // 创建一个新的HTTP客户端
    client := &http.Client{}
    // 创建一个新的请求
    //req, err := http.NewRequest("POST", *signStr, bytes.NewBuffer([]byte(partInfo)))
    req, err := http.NewRequest("POST", *signStr, bytes.NewBufferString(partInfo))
    // 请求消息头中增加 "Content-Type":"application/xml"
    req.Header.Set("Content-Type", "application/xml")
    // 发送请求
    resp, err := client.Do(req)
    if err != nil {
        fmt.Println("Error sending request:", err)
        panic(err)
    }
    defer func(Body io.ReadCloser) {
        err := Body.Close()
        if err != nil {
            panic(err)
        }
    }(resp.Body)
    // 读取响应体
    body, _ := io.ReadAll(resp.Body)
    // 打印响应体
    fmt.Println(string(body))
    fmt.Println("合并分段 mergeUploadedPart end\n")
}

/*
12.确认上传完成
*/
func confirmUploaded(client *vod.VodClient, assetResponse *model.CreateAssetByFileUploadResponse) {
    fmt.Println("确认上传完成 confirmUploaded start")
    // 设置请求参数
    request := &model.ConfirmAssetUploadRequest{}
    request.Body = &model.ConfirmAssetUploadReq{
        Status: model.GetConfirmAssetUploadReqStatusEnum().CREATED,
        AssetId: *assetResponse.AssetId,
    }
    // 发送请求
    response, err := client.ConfirmAssetUpload(request)
    if err == nil {
        fmt.Print(response)
    } else {
        fmt.Println(err)
        panic(err)
    }
    fmt.Println("上传完成, assetId:", *response.AssetId)
}

// InitiateMultipartUploadResult 初始化分段上传返回的xml结构体
type InitiateMultipartUploadResult struct {
    Bucket string `xml:"Bucket"`
    Key string `xml:"Key"`
}

```



```

    UploadId string `xml:"UploadId"`
}

// ListPartsResult 查询已上传段返回的xml结构体
type ListPartsResult struct {
    Bucket      string `xml:"Bucket"`
    Key         string `xml:"Key"`
    UploadId    string `xml:"UploadId"`
    NextPartNumberMarker int    `xml:"NextPartNumberMarker"`
    Parts       []Part `xml:"Part"`
}

// Part 查询已上传段返回的xml listPartsResult中part结构体
type Part struct {
    Etag      string `xml:"ETag"`
    PartNumber int   `xml:"PartNumber"`
}

```

19.7.6 PHP

分段上传PHP语言的示例代码，如下所示：

```

<?php
namespace HuaweiCloud\SDK\Vod\V1\Model;

use DOMDocument;
use DOMException;
use Exception;

require_once "vendor/autoload.php";
use HuaweiCloud\SDK\Core\Auth\BasicCredentials;
use HuaweiCloud\SDK\Core\Http\HttpConfig;
use HuaweiCloud\SDK\Vod\V1\VodClient;

// 区域
const region_north1 = 'cn-north-1';
const region_north4 = 'cn-north-4';
const region_east2 = 'cn-east-2';

// ak/sk
const ak = "";
const sk = "";

const projectId = "";

const endpoint = "https://vod." . region_north4 . ".myhuaweicloud.com";

// 设置缓冲区大小，每次读取文件分段的大小，根据情况自行设定
// 1M
const bufferSize = 1024 * 1024;

main();

/**
 * 分段上传示例
 */
function main(): void
{
    // 本地要上传的媒资路径
    $filePath = "";

    // 上传媒资文件
    partUpload($filePath);
}

/**
 * 分段上传
 * @param $filePath 传文件的本地路径
 * @return void
 */

```

```

*/
function partUpload(string $filePath)
{
    $handle = fopen($filePath, "r");
    if(!$handle){
        throw new Exception("无法打开文件!");
    }
    try {
        $fileName = basename($filePath);

        // 此处仅以MP4文件示例，其他格式可以参考官网说明
        $fileType = "MP4";
        $fileContentType = "video/mp4";

        // 1.初始化鉴权并获取vodClient
        $vodClient = createVodClient();
        echo "开始创建媒资: " . $fileName . "\n";
        // 2.创建点播媒资
        $assetResponse = createAsset($vodClient, $fileName, $fileName, $fileType);
        // 3.获取初始化上传任务授权
        $initAuthResponse = initPartUploadAuthority($vodClient, $assetResponse, $fileContentType);
        // 4.初始化上传任务
        $uploadId = initPartUpload($initAuthResponse->getSignStr(), $fileContentType);
        // 文件分段计数
        $partNumber = 1;

        // 7.读取文件内容，循环5-6步上传所有分段
        while (!feof($handle)) {
            $content = fread($handle, bufferSize);
            // 先md5,再base64
            $md5Hash = md5($content);
            echo "md5:". $md5Hash ."\n";
            $hexString = hex2bin($md5Hash);
            $contentMd5 = base64_encode($hexString);
            echo "该文件第" . $partNumber . "段contentMd5:" . $contentMd5 . "\n";

            // 5.获取上传分段的授权
            $partUploadAuthorityResponse = getPartUploadAuthority($vodClient, $fileContentType,
                $assetResponse, $contentMd5, $uploadId, $partNumber);
            // 6.上传分段
            uploadPartFile($partUploadAuthorityResponse->getSignStr(), $content, $contentMd5);

            //段号自增
            $partNumber++;
        }
        fclose($handle);

        // 8.获取已上传分段的授权
        $listPartUploadAuthorityResponse = listUploadedPartAuthority($vodClient, $assetResponse, $uploadId);
        // 9.获取已上传的分段
        $partInfo = listUploadedPart($listPartUploadAuthorityResponse->getSignStr());
        // 10.获取合并段授权
        $mergePartUploadAuthorityResponse = mergeUploadedPartAuthority($vodClient, $assetResponse,
            $uploadId);
        // 11.合并上传分段
        mergeUploadedPart($mergePartUploadAuthorityResponse->getSignStr(), $partInfo);
        // 12.确认媒资上传
        confirmUploaded($vodClient, $assetResponse);

        echo "创建媒资结束 assetId:" . $assetResponse->getAssetId() . "\n";
    } catch (Exception $e) {
        echo $e;
    }
}

/**
 * 1.构建鉴权
 * @return VodClient

```

```

*/
function createVodClient(): VodClient
{
    $credentials = new BasicCredentials(ak, sk, projectId);
    $config = HttpConfig::getDefaultConfig();
    $config->setIgnoreSslVerification(true);

    $client = VodClient::newBuilder(new VodClient)
        ->withHttpConfig($config)
        ->withEndpoint(endpoint)
        ->withCredentials($credentials)
        ->build();
    return $client;
}

/**
 * 2.创建媒资
 * @param VodClient $vodClient
 * @param string $videoName
 * @param string $title
 * @param string $videoType
 * @return CreateAssetByFileUploadResponse $response
 */
function createAsset(VodClient $vodClient, string $videoName, string $title, string $videoType):
CreateAssetByFileUploadResponse
{
    $request = new CreateAssetByFileUploadRequest();
    $body = new CreateAssetByFileUploadReq();
    $body->setVideoName($videoName);
    $body->setTitle($title);
    $body->setVideoType($videoType);
    $request->setBody($body);
    $response = $vodClient->createAssetByFileUpload($request);
    echo $response . "\n";
    return $response;
}

/**
 * 3.获取初始化上传任务授权
 * @param VodClient $client
 * @param CreateAssetByFileUploadResponse $assetResponse
 * @param string $fileContentType
 * @return ShowAssetTempAuthorityResponse
 */
function initPartUploadAuthority(VodClient $client, CreateAssetByFileUploadResponse $assetResponse,
string $fileContentType): ShowAssetTempAuthorityResponse
{
    echo "获取初始化上传任务授权 initPartUploadAuthority start\n";
    $request = new ShowAssetTempAuthorityRequest();
    // 设置参数
    $request->setHttpVerb("POST");
    $request->setBucket($assetResponse->getTarget()->getBucket());
    $request->setObjectKey($assetResponse->getTarget()->getObject());
    $request->setContentType($fileContentType);
    // 发送初始化请求
    $response = $client->showAssetTempAuthority($request);
    echo "获取初始化上传任务授权 end; response:" . $response . "\n\n";
    return $response;
}

/**
 * 4.初始化分段上传
 * @param string $signStr
 * @param string $contentType
 * @return string|null
 */
function initPartUpload(string $signStr, string $contentType): string
{
    echo "初始化分段上传 initPartUpload start\n";
}

```

```

// 初始化cURL会话
$ch = curl_init();
//设置cURL选项
curl_setopt($ch, CURLOPT_URL, $signStr);
curl_setopt($ch, CURLOPT_POST, true);
curl_setopt($ch, CURLOPT_HTTPHEADER, array(
    "Content-Type: " . $contentType // 设置请求头
));
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, false); //验证curl对等证书(一般只要此项)
curl_setopt($ch, CURLOPT_SSL_VERIFYHOST, false); //检查服务器SSL证书中是否存在一个公用名
curl_setopt($ch, CURLOPT_SSLVERSION, 0); //传递一个包含SSL版本的长参数。

// 执行cURL会话
$response = curl_exec($ch);
if (curl_errno($ch)) {
    echo 'cURL Error: ' . curl_error($ch);
} else {
    // 打印响应内容
    echo $response;
}

echo $response . "\n";
$initDom = new DOMDocument();
$initDom->loadXML($response);
curl_close($ch);
$UploadIdNode = $initDom->getElementsByTagName("UploadId");
$uploadId = $UploadIdNode->item(0)->nodeValue;
echo "初始化分段上传 initPartUpload end; UploadId:" . $uploadId . "\n\n";
return $uploadId;
}

/**
 * 5.获取分段上传授权
 * @param VodClient $client
 * @param string $fileContentType
 * @param CreateAssetByFileUploadResponse $assetResponse
 * @param string $contentMd5
 * @param string $uploadId
 * @param int $partNumber
 * @return ShowAssetTempAuthorityResponse
 */
function getPartUploadAuthority(VodClient $client, string $fileContentType,
    CreateAssetByFileUploadResponse $assetResponse,
    string $contentMd5, string $uploadId, int $partNumber):
    ShowAssetTempAuthorityResponse
{
    echo "获取分段上传授权 getPartUploadAuthority start; partNumber:" . $partNumber . "\n";
    $request = new ShowAssetTempAuthorityRequest();
    $request->setHttpVerb("PUT");
    $request->setBucket($assetResponse->getTarget()->getBucket());
    $request->setObjectKey($assetResponse->getTarget()->getObject());
    $request->setContentType($fileContentType);
    $request->setContentMd5($contentMd5);
    $request->setUploadId($uploadId);
    $request->setPartNumber($partNumber);
    $response = $client->showAssetTempAuthority($request);
    echo "获取分段上传授权 getPartUploadAuthority end; partNumber:" . $partNumber . "\n";
    return $response;
}

/**
 * 6.上传分段
 * @param string $signStr
 * @param string $fileByte
 * @param string $contentMd5
 * @return void
 * @throws Exception
 */

```

```

function uploadPartFile(string $signStr, $fileByte, string $contentMd5)
{
    echo "上传分段 uploadPartFile start\n";
    // echo $fileByte . "\n";
    // 初始化cURL会话
    $ch = curl_init();
    // 设置cURL选项
    curl_setopt($ch, CURLOPT_URL, $signStr);
    curl_setopt($ch, CURLOPT_CUSTOMREQUEST, 'PUT');
    curl_setopt($ch, CURLOPT_POSTFIELDS, $fileByte);
    curl_setopt($ch, CURLOPT_HTTPHEADER, array(
        "Content-MD5: " . $contentMd5,
        "Content-Type: application/octet-stream"
    ));
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);

    curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, false); //验证curl对等证书(一般只要此项)
    curl_setopt($ch, CURLOPT_SSL_VERIFYHOST, false); //检查服务器SSL证书中是否存在一个公用名
    curl_setopt($ch, CURLOPT_SSLVERSION, 0); //传递一个包含SSL版本的长参数。
    // 执行cURL会话
    $response = curl_exec($ch);
    if (curl_errno($ch)) {
        echo 'cURL Error: ' . curl_error($ch);
        // 打印响应内容
        echo $response . "\n";
        throw new Exception("上传失败!");
    }
    // 关闭会话
    curl_close($ch);
    echo "上传分段 uploadPartFile end;\n\n";
}

/**
 * 8.获取列举已上传段的授权
 * @param VodClient $client
 * @param CreateAssetByFileUploadResponse $assetResponse
 * @param string $uploadId
 * @return ShowAssetTempAuthorityResponse
 */
function listUploadedPartAuthority(VodClient $client, CreateAssetByFileUploadResponse $assetResponse,
string $uploadId):ShowAssetTempAuthorityResponse
{
    echo "获取列举已上传段的授权 listUploadedPartAuthority start\n";
    $request = new ShowAssetTempAuthorityRequest();
    $request->setHttpVerb("GET");
    $request->setBucket($assetResponse->getTarget()->getBucket());
    $request->setObjectKey($assetResponse->getTarget()->getObject());
    $request->setUploadId($uploadId);
    $response = $client->showAssetTempAuthority($request);
    echo "列举已上传段的授权 listUploadedPartAuthority end; response: " . $response . "\n";
    return $response;
}

/**
 * 9.查询已上传的分段
 * @param string $signStr
 * @return false|string
 * @throws DOMException
 */
function listUploadedPart(string $signStr): false|string
{
    echo "查询已上传的分段 listUploadedPart start\n";
    $partNumberMarker = 0;
    $compDom = new DOMDocument();
    $root = $compDom->createElement("CompleteMultipartUpload");
    $compDom->appendChild($root);
    while(true)
    {
        // 初始化cURL会话

```

```

$ch = curl_init();
// 设置cURL选项
curl_setopt($ch, CURLOPT_URL, $signStr . "&part-number-marker=" . $partNumberMarker);
curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, false); //验证curl对等证书(一般只要此项)
curl_setopt($ch, CURLOPT_SSL_VERIFYHOST, false); //检查服务器SSL证书中是否存在一个公用名
curl_setopt($ch, CURLOPT_SSLVERSION, 0); //传递一个包含SSL版本的长参数。

// 执行cURL会话
$response = curl_exec($ch);
if (curl_errno($ch)) {
    echo 'cURL Error: ' . curl_error($ch);
} else {
    // 打印响应内容
    echo $response . "\n";
}
// 关闭cURL会话
curl_close($ch);
$listDom = new DOMDocument();
$listDom->loadXML($response);
$nextPartNumberMarkerNode = $listDom->getElementsByTagName("NextPartNumberMarker");
$partNodes = $listDom->getElementsByTagName("Part");
foreach($partNodes as $partNode)
{
    $partNumberNode = $partNode->getElementsByTagName("PartNumber");
    $partNumber = $partNumberNode->item(0)->nodeValue;
//    echo $partNumber . "\n";
    $eTagNode = $partNode->getElementsByTagName("ETag");
//    $eTag = $eTagNode->item(0)->nodeValue;
//    echo $eTag . "\n";
    try{
        $part = $compDom->createElement("Part");
        $part->appendChild($compDom->createElement("PartNumber", $partNumber));
        $part->appendChild($compDom->createElement("ETag", $eTag));
        $root->appendChild($part);
    } catch (DOMException $e){
        echo $e->getMessage();
    }
}
$partNumberMarker = $nextPartNumberMarkerNode->item(0)->nodeValue;
if($partNumberMarker % 1000 != 0 || $partNumberMarker == 0)
{
    break;
}
}
$partInfo = $compDom->saveXML();
echo $partInfo . "\n";
echo "查询已上传的分段 listUploadedPart end\n\n";
return $partInfo;
}

/**
 * 10.获取合并段授权
 * @param VodClient $client
 * @param CreateAssetByFileUploadResponse $assetResponse
 * @param string $uploadId
 * @return ShowAssetTempAuthorityResponse
 */
function mergeUploadedPartAuthority(VodClient $client, CreateAssetByFileUploadResponse $assetResponse,
string $uploadId):ShowAssetTempAuthorityResponse
{
    echo "获取合并段授权 mergeUploadedPartAuthority start\n";
    $request = new ShowAssetTempAuthorityRequest();
    $request->setHttpVerb("POST");
    $request->setBucket($assetResponse->getTarget()->getBucket());
    $request->setObjectKey($assetResponse->getTarget()->getObject());
    $request->setUploadId($uploadId);
    $response = $client->showAssetTempAuthority($request);
    echo "获取合并段授权 mergeUploadedPartAuthority end; response:" . $response . "\n";
}

```

```

    return $response;
}

/**
 * 11.合并分段
 * @param string $signStr
 * @param string $partInfo
 * @return void
 */
function mergeUploadedPart(string $signStr, string $partInfo)
{
    echo "合并分段 mergeUploadedPart start\n";
    // 初始化cURL会话
    $ch = curl_init();
    curl_setopt($ch, CURLOPT_URL, $signStr);
    curl_setopt($ch, CURLOPT_POST, true);
    curl_setopt($ch, CURLOPT_POSTFIELDS, $partInfo);
    // 请求消息头中增加"Content-Type",值设置为"application/xml"
    curl_setopt($ch, CURLOPT_HTTPHEADER, array(
        "Content-Type: application/xml"
    ));
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
    curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, false); //验证curl对等证书(一般只要此项)
    curl_setopt($ch, CURLOPT_SSL_VERIFYHOST, false); //检查服务器SSL证书中是否存在一个公用名
    curl_setopt($ch, CURLOPT_SSLVERSION, 0); //传递一个包含SSL版本的长参数。
    // 执行cURL会话
    $response = curl_exec($ch);
    if (curl_errno($ch)) {
        echo 'cURL Error: ' . curl_error($ch);
    } else {
        // 打印响应内容
        echo $response . "\n";
    }
    // 关闭会话
    curl_close($ch);
    echo "合并分段 mergeUploadedPart end\n";
}

/**
 * 12.确认上传完成
 * @param VodClient $client
 * @param CreateAssetByFileUploadResponse $assetResponse
 * @return void
 */
function confirmUploaded(VodClient $client, CreateAssetByFileUploadResponse $assetResponse): void
{
    echo "确认上传完成 confirmUploadedPart start\n";
    $request = new ConfirmAssetUploadRequest();
    $body = new ConfirmAssetUploadReq();
    $body->setStatus("CREATED");
    $body->setAssetId($assetResponse->getAssetId());
    $request->setBody($body);
    $response = $client->confirmAssetUpload($request);
    echo "上传完成 confirmUploadedPart end; response:" . $response . "\n";
}

```

20 修订记录

表 20-1 修改记录

修改时间	修改说明
2024-11-15	第十二次正式发布 本次变更如下： 新增“转码产物管理”接口组。
2024-10-15	第十一次正式发布 本次变更如下： <ul style="list-style-type: none"> 新增“修改媒资降冷粒度”接口。 新增“查询媒资降冷”接口。 新增Python和Go语言的分段上传代码示例。
2024-08-15	第十次正式发布 本次变更如下： “转码模板管理”接口组新增参数preset、max_iframes_interval和hls_audio_separate。
2024-06-18	第九次正式发布 本次变更如下： 新增Java、JavaScript语言的分段上传代码示例。
2023-12-21	第八次正式发布 本次变更如下： 新增2个接口：查询媒资日播放统计数据、修改媒资文件在OBS的存储模式。
2023-07-14	第七次正式发布 本次变更如下： <ul style="list-style-type: none"> 查询CDN统计信息接口中“values”字段新增单位说明。 优化查询源站统计信息接口的功能介绍。

修改时间	修改说明
2023-05-23	<p>第六次正式发布</p> <p>本次变更如下：</p> <ul style="list-style-type: none"> • 示例1：媒资上传（20M以下）中新增音频文件格式对应的请求头填写规则说明。 • 优化“示例2：媒资分段上传（20M以上）”章节内容。 • 获取分段上传授权接口中“content_type”字段新增文件格式说明。
2023-05-06	<p>第五次正式发布</p> <p>本次变更如下：</p> <ul style="list-style-type: none"> • 新增终端节点（Endpoint）信息。 • 查询CDN统计信息接口新增查询统计数据存在1小时时间差的描述。
2023-03-30	<p>第四次正式发布</p> <p>本次变更如下：</p> <p>创建媒资：OBS转存方式接口中“video_type”字段新增RMVB和WEBM视频格式。</p>
2023-03-08	<p>第三次正式发布</p> <p>本次变更如下：</p> <p>新增多字幕封装接口。</p>
2022-11-23	<p>第二次正式发布</p> <p>本次变更如下：</p> <ul style="list-style-type: none"> • 修改创建自定义转码模板接口中可选参数“name”、“quality_info_list”、“common”为必选参数。 • 更新修改转码模板接口中可选参数“group_id”、“quality_info_list”、“common”为必选参数。 • 修改创建转码模板组集合接口中可选参数“name”、“template_group_list”为必填参数。 • 更新修改转码模板组集合接口中可选参数“collection_id”为必填参数。
2022-10-12	<p>第一次正式商用发布。</p>