百度智能云

MCT 文档

目录

功能发布记录

发布时间	功能分类	功能名称	相关文档	发布地域
2023-05	新功能	• 新增感知编码转码类型 • 新增版权保护：数字水印添加和检测功能	-	华北-北京、华南-广州、华东-苏州
2023-04	服务端SDK	• 新增Golang-SDK支持	-	华北-北京、华南-广州、华东-苏州
2022-12	新功能	• 视频质量检测支持存储检测帧图片； • 视频质量检测支持设置任务级别的回调； • 视频质量检测新增任务重新运行的API接口； • 视频质量检测任务详情和结果导出的时间戳精确到毫秒； • 视频质量检测算子参数新增，支持个性化设置各个算子的检测间隔和持续时长参数； • 视频质量检测算子拆分，亮度算子拆分为黑屏、白屏、过亮、过暗4种算子，色度算子拆分为纯色和偏色算子各5种（红、黄、绿、蓝、紫）； • 视频质量检测算子新增，新增模糊边缘、静态边缘、花屏、彩条、块效应、场效应、静音、音量过大、音量过低、声音间断算子； • 视频质量检测算子下线，下线了滚动条纹算子。	• 视频质量检测任务接口 • 视频质量检测模板接口	华北-北京、华南-广州、华东-苏州
		• 开放最大I帧间隔（GOP）设置； • 支持闲时转码，以更低的价格，获得专业稳定的视频处理服务；		

2022-06	新功能	• 支持即时转码，可在播放请求时，按需下发转码任务，大大节省了存储费用； • 系统模板进行了全面更新，按不同使用场景分为：普通转码、自适应码流、智能编码（CAE）、老片修复、画质提升； • 支持Dash自适应码流，更好的适应播放端的网络情况，提供更流畅的播放体验； • 支持HDR视频的输入和输出，支持格式包括：HLG、HDR10、vivid； • 支持BD265编码器，可在编码标准里选择，相比开源的X265，降低30%+码率，提升2-4倍速。	视频转码模板接口	华北-北京、华南-广州、华东-苏州
2022-05	新功能	• 新增音视频质量检测能力，支持10种音频和画面问题检测算子； • 新增老片修复、画质提升等智感超清能力，通过智能去噪、去划痕、增强、超分、SDRtoHDR等AI算子，大幅提升视频画质； • 支持wanos 全景声音频标准，提供了更具沉浸感更立体的声音体验。	视频质量检测任务接口	华北-北京、华南-广州、华东-苏州
2021-02	名称变更	音视频转码MCT（Multimedia Cloud Transcoder）更名为音视频处理MCP（Multimedia Cloud Processing）	-	华北-北京、华南-广州、华东-苏州
2020-07	新功能	• 新增概览数据统计，包括近一个月的转码时长、转码任务数、缩略图张数。 • 内置转码模板更新，新的模板都是16：9的比例，每个清晰度下保留一个通用的优化模板。 • 水印模板优化，支持设置水印文件大小；水印位置支持设置绝对值和百分比 • 新增通知管理，支持对通知地址的新增、修改、删除管理。 • console交互优化	• 创建转码模板 • 创建水印模板 • 通知管理	华北-北京、华南-广州、华东-苏州
		新增缩略图模板管理，支持对缩略图任务模板化管理。支持	缩略图模板接	华北-北京、华

2020-06	新功能	新增、修改、删除管理。	口	南-广州、华东-苏州
2019-07	新功能	• 新增HLS、MP4等主流媒体格式多码率无缝切换功能 • 新增缓冲区时长设置功能 • 新增cuid设置接口	• iOS播放器接口速查 • Andriod播放器 接口速查	华北-北京、华南-广州、华东-苏州
2019-6	新功能	首发Anroid SDK高级播放器	播放器SDK Android平台	华北-北京、华南-广州、华东-苏州
2019-06	新功能	• 智能超清1.0，根据内容复杂度，动态分配最优码率，保障画面质量的前提下降低码率，从而降低带宽和存储成本 • 智感超清2.0， 通过画面主观增强，调节饱和度、亮度、对比度等，并进行画质修复，去除画面伪影、毛刺、马赛克等，在码率不变的前提下，提高画面质量	• 功能特性 • 使用限制	华北-北京
2019-03	新功能	• 任务队列改造为running任务为20*5（最多5个队列，最大总容量100），pending排队任务书最大为1000，更好的应对用户任务激增	创建任务队列	华北-北京、华南-广州、华东-苏州
2018-12	新功能	MCT接入IAM系统，只支持系统策略	-	华北-北京、华南-广州、华东-苏州
2018-11	新功能	• 上线精彩动图智能提取，根据视频内容的相关性及画面的优美程度，提取最精彩的片段生成动图，可作为视频封面，提高运营效果	缩略图任务接口	华北-北京、华南-广州、华东-苏州
2018-10	新功能	• 支持单段视频剪辑及叠加多水印后，进行多段视频拼接 • 支持黑边自动检测或指定黑边区域，进行视频或者图片黑边剪裁 • 支持文本、字幕、图片、音频、视频等多格式的水印叠加，格式支持：JPG、PNG、APNG、BMP、PBM、GIF、TIF、MOV、MP4、MP3、srt等	视频转码任务接口	华北-北京、华南-广州、华东-苏州
2018-10	新功能	• 在创建转码模板时可设置crf恒定质量因子（0-51） • 横竖模板自适应，当原视频为竖形时，自动调整模板的宽小于高，保证缩放比最小，反之亦然 • 在创建转码模板时可将音频音量统一化为同样大小，可设置静音、调节音量大小 • 上线智感超清转码CAE，根据画面复杂度及场景动态分配	视频转码模板接口	华北-北京、华南-广州、华东-苏州

		转码码率，以最低的码率获得最高清的画面质量，大大节省了带宽成本 • 创建转码模板时，封装格式支持dash，可用于iOS和Android端的播放，可实现码率自适应，提高首屏加载率		
2018-2	上线	首次发布	产品简介	华北-北京、华南-广州、华东-苏州

产品描述

产品简介

音视频处理MCP（Multimedia Cloud Processing）针对海量媒资提供了高效、智能、稳定的媒体处理服务，包括：标准转码、智感超清、智能视频处理与生产、智能视频质量检测、媒体版权保护（数字水印）、播放器。全方位满足音视频处理及多终端流畅高清播放需求。

MCP标准编码提供百度自研编码器，三种编码标准：H264、H265、AV1。相比x264/265/AV1，码率节省**50%+**，速度提升**2倍+**；

MCP智感超清，结合了多种视觉AI与编码技术，通过AI模型深度学习，根据内容场景及复杂度，智能调节编码参数，并优化主观视觉体验，以更小的码率获得了更好的编码质量，包括：智能超分、画质增强、SDRtoHDR等；

MCP视频处理与生产基于视觉AI与编码技术，支持视频进行批量编辑和处理，包括：智能横转竖、智能去水印、智能去字幕等；

MCP视频质量检测基于视频文件在录制、传输和存储过程中可能会出现数据损坏从而导致视频数据中出现一些缺陷的需求场景考虑，通过智能算法，对画面模糊、花屏、噪声等问题进行分析和评估，帮助用户快速定位问题，从而进行必要调整。

MCP媒体版权保护主要提供数字水印解决方案，支持添加和检测不可见的数字水印，可追踪版权来源。为用户带来极致的主观体验，大幅提升视频和图片的安全性。

MCP是基于百度智能云构建，以公有云的方式提供服务，开通即可使用，按量后付费。我们提供SLA服务等级协议，全方位保障服务的稳定性、可用性。通过智能调度和资源弹性伸缩，更好的帮助您应对业务变化，具备高质量、高效率、强稳定、低成本的特性。

名词解释

队列（Pipeline）

通过队列，用户可以更灵活地管理转码/缩略图任务。当用户创建一个任务时，用户必须为该任务指定所属的队列。

模板（Preset）

模板是对于视频资源在做转码时所需定义的参数集合。用户可以更简便的将一个模板应用于一个和多个视频的转码任务，以输出目标视频文件。

任务（Job）

任务是音视频转码中最基本的执行单元，每个任务将一个原始的音视频资源转码成目标规格的音视频资源。因此，任务和转码的目标是一一对应的，也就是说如果用户需要将一个原始视频规格转换成三种目标规格，比如从AVI格式转码成FLV/MP4/HLS格式，那么用户将会需要创建三个任务。

消息通知

通过选择消息通知方式，可以使用户随时了解任务的状态。不论处理成功还是失败，在处理任务结束时均会触发消息通知。在创建任务队列时需要配置通知接口，目前仅支持HTTP POST通知。

功能特性

视频处理将音视频文件转码为不同分辨率、不同格式的文件，以满足不同网络带宽、不同终端设备的用户需求。核心能力特性如下所示：

- 转码格式：覆盖主流的视频转码格式H264、H265、AV1。
- 标准音视频转码：丰富的视频和音频编码参数支持，支持多种视频格式、视频分辨率、不同码率等。
- 视频处理：支持去除/添加水印、横转竖、去黑边等视频处理能力。
- 视频截图：支持多种视频截图能力，满足不同场景的需求，截取精美封面，提高视频点击率。
- 视频加密：版权保护，防盗播。
- 智感超清：通过视频AI的能力，提高视频的清晰度，降低视频的码率。
- 极速转码：音视频分离，动态分片技术提高转码速度，最高可达50倍速。
- 视频质检：支持亮度、偏色、模糊、噪声、马赛克、花屏等视频质量问题的检测，支持音量过高/过低、声音间断等音频质量问题的检测。
- 数字水印：支持向视频和图片中嵌入肉眼不可见的文字或图片，且对视频质量影响小，对常见攻击具有一定抵抗能力。在发生版权纠纷时，可通过数字水印提取服务提取水印内容，证明版权归属。

标准音视频转码

类别	说明
输入格式	· 封装格式：MP4、FLV、MOV、M3U8、3GP、AVI、MPG、ASF、WMV、MKV、TS、WebM、MXF； · 视频编码格式：H.264/AVC、H.265/HEVC、AV1、MPEG-1、MPEG-2、MPEG-4、MJPEG、VP8、VP9、Quicktime、RealVideo、Windows Media Video ； · 音频编码格式：AAC、AC-3、ADPCM、AMR、DSD、MP1、MP2、MP3、PCM、RealAudio、Windows Media Audio
输出格式	· 视频封装格式：MP4 、FLV、HLS、MP3、M4A、ADAPTIVE_HLS、PCM、DASH、MXF、TS、MOV、AVI、MKV、OGG、DPX、WEBM； · 音频封装格式：MP3、MP4、OGG、FLAC、m4a； · 图片封装格式：JPG、PNG、GIF、WEBP； · 视频编码格式：H.264/AVC、 H.265/HEVC、BD264、BD265、BDAV1； · 音频编码格式：MP3、AAC、VORBIS、FLAC、DTS、杜比全景声
码率控制	· 支持CBR、VBR、CRF、CAE、CQE等码率控制方式的设置.(针对同一视频，CAE可节省编码码率，VMAF图像质量客观指标优于VBR。)
输出视频参数设置	· 支持分辨率、码率、帧率、编码Profile/Level设置； · 支持GOP长度（I帧间隔）和最大B帧数设置； · 支持音量均衡化，避免音量忽高忽低。
视频多功能处理	· 支持视频拼接：支持设置起始时间、持续时长，单位为秒，多个视频拼接为1个； · 叠加字幕：支持原视频添加.srt字幕文件； · 支持视频剪辑：支持设置输出视频起始时间、持续时长； · 支持3种模式的视频加密：fixed 固定密钥加密，使用用户指定的密钥对视频进行加密，此时需要aesKey； 开放密钥，系统自动生成加密密钥，密钥公开，不设访问控制； 系统自动生成加密密钥，密钥设有访问控制，绑定播放器，安全性比较高，推荐。

自适应码流

自适应码流ABR（Adaptive Bitrate Streaming）的特点是将视频内容分片成一系列片段，每个片段有不同码率，播放端根据当前带宽，动态选择最合适的码率进行播放。当前MCP支持将视频转成DASH自适应码流格式，当内容由MPEG-DASH客户端播放时，客户端根据比特率自适应（ABR）算法和自身性能，自动选择具有最适宜比特率的片段，可以及时下载该片段进行播放而不会造成停顿或在播放中重新缓冲事件，支持在不同码率之间无缝切换。

能力	说明
支持格式	DASH
支持编码	支持以上所述编码能力：H.264、H.265
智能分辨率切换	播放器能够根据当前带宽，动态选择最合适的分辨率播放。
起播超低延时	自适应码流包含多种分辨率，播放器通常从低分辨率开始起播，起播速度快。
切换过程零卡顿	自适应码流中各个分辨率做到帧对齐，切换过程中零卡顿。

极速转码

极速转码包含普通极速转码和智能极速转码。 普通极速转码通过音视频分离的技术，对音频、视频分开转码，减少视频merge时间，对于长视频转码倍速可达10倍速。 智能极速转码在普通极速转码的基础上，根据输入视频/输出模板的属性（Codec、B帧、帧率、码率、分辨率等），通过AI模型预测分片策略，转码速度最高可达50倍速。

即时转码

MCP提供了业界全新的视频即时转码接口，支持通过设置url参数的方法实时转换视频的码率，实现h264，h265两种码率的互转。在直播录流时，将存储的文件编码格式从h264转为h265，降低了存储成本；而在用户请求回看视频时，支持从h265快速转为h264，适配大部分播放器。满足用户在直播录流和回放直播视频的场景节省存储成本的同时又能保证播放器流畅的需求。

感知编码

基于百度强大的编码器BD264（H.264标准）、BD265（H.265标准），结合AI图像处理、CAE感知编码、ROI等技术，感知编码实现点播、直播业务以更低码率传输的同时，给用户带来更高清的画质体验。

AI视频处理

类别	应用场景	功能说明
智能横转竖	通过目标检测算法识别出画面中的重要人物和精彩区域，动态调整窗口位置的方式将视频从横屏（16:9）转成竖屏（9:16）。	--
叠加明水印	应用于对视频版权所属可视化，降低信息泄漏风险。	· 支持添加水印类型：图片、文本（暂未开放）； · 支持自定义水印图片位置、大小； · 格式包括：jpg、png、apng、gif、webp、mov、mp4等，支持设置显示位置和起始时间
智能去水印	对视频内的显性水印进行智能去除，应用于视频搬运，二次编辑和分发场景。	· 支持两种去水印的形式：智能去水印、手动去除水印； · 支持去除的水印形态包括：静态Logo、固定位置的动态Logo、半透明Logo、镂空Logo
智能去字幕	支持自动识别视频底部字幕的区域并去除，应用于视频搬运，二次编辑和分发场景。	· 支持设置去字幕的区域（x、y、width、height），支持自动识别水印并去除叠加字幕：支持原视频添加.srt字幕文件； · 支持设置裁剪黑边后的有效画面区域（x、y、width、height），支持自动检测和去除黑边区域；支持自动去除片头黑帧。
智能去抖动	支持对由于录制设备问题导致视频有大幅度抖动现象的识别，并自动缓解抖动现象。	--
智能封面	百度智能封面帮助用户对视频高光帧或高光片段提取静态和动态图片，可用于在视频分发平台展示封面。	
智能去黑边	解决由于视频二次分发时由于设备尺寸的变化所产生画面冗余的黑边现象。一方面可以提升用户的观感，另一方面也可以节省码率，减少文件体积。	· 支持设置裁剪黑边后的有效画面区域（x、y、width、height），支持自动检测和去除黑边区域；支持自动去除片头黑帧。
智能绿幕抠像	针对绿幕录制场景视频，支持自动扣除人像，生成透明通道背景的视频.webp。生成的视频添加任意背景。	--
提取元信息	针对视频内容属性等描述进行提取，让用户快速了解到视频的信息从而做出正确的业务判断。	· 支持提取文件信息：文件大小、文件时长、容器格式、文件类型、MD5值； · 支持提取视频信息：编码标准、分辨率（宽/高）、码率、帧率 ； · 支持提取音频信息：编码标准、声道、采样率、码率。

智感超清

类别	说明
画质提升	以人眼主观体验最好为目标，结合AI图像处理技术，通过智能的细节增强、色彩增强、超分辨率、插帧、SDRtoHDR等能力大幅提升画质。
老片修复	针对老片或过度压缩的视频，去除抖动、划痕、噪点、马赛克等，提高画面清晰度。
智能插帧	对于30帧/秒及以内的普通帧率视频，生成60帧/秒甚至120帧/秒的高帧率版本，提高画面流畅度，一般配合超分使用。
超分辨率	利用深度学习模型，提升视频画面的细节，将低分辨率重建至高分辨率，例如：SD转HD 、2K转4K等。

视频质检

类别	说明
亮度	检测视频超出人眼舒适范围的过亮、过暗的片段（包含全黑屏、白屏）。
偏色	检测视频偏色的片段，包含偏红、黄、蓝、绿、紫5种偏色。
纯色屏	检测视频画面的纯色屏，包含：黑、白、红、黄、蓝、绿、紫。
模糊	检测视频由于聚焦不当镜头损坏等因素引起的视野主体部分的图像模糊的片段。
噪声	检测视频图像中混有呈带状、波纹、网状等带有周期性的叠加噪声的片段。
马赛克	检测由于录制视频时出现网络异常、卫星信源干扰等原因导致滚动条纹的片段。
滚动条纹	检测由于录制视频时摄像设备和显示器的屏幕刷新频率不同步或由于摩尔条纹的干扰视频导致出现滚动条纹的片段。
抖动	检测视频由于录制设备晃动导致画面抖动的片段。
静帧	检测视频由于网络信号不稳定导致画面定格/冻结的片段。
黑边	检测视频画面中出现黑边（包含黑色、其他纯色和高斯模糊边界）的片段。
花屏	检测视频在直播推流过程中连接线接触不良、信号源传输错误等问题造成录制的视频出现花屏的片段。
彩条	检测视频在直播推流过程中连接线接触不良、信号源传输错误等问题造成录制的视频出现彩条的片段。
块效应	检测视频随着码率的降低，在块的边界会出现不连续、形成重建图像的明显缺陷的片段。
场效应	检测视频后期处理阶段由于压缩导致图像运动处出现交错行的片段。
静音	检测音频出现非预期的静音的片段。
音量过高/过低	检测音频出现超出人耳舒适范围的音量过高/过低的片段。
声音间断	检测由于录制过程中信号源输入不稳定导致音频断断续续的片段。

数字水印

视频数字水印功能特性

类别	说明
被检原视频格式	MP4、FLV、MOV、M3U8、3GP、AVI、MPG、ASF、WMV、MKV、TS、WebM、MXF
水印类型	支持图片、文本两种类型水印嵌入
抗攻击性	·能够抵抗一定程度的转码攻击 ·能够抵抗一定程度的画面缩放攻击 ·能够抵抗一定程度的画面遮挡攻击 ·能够抵抗亮度/对比度变化的攻击 ·支持对录屏视频进行数字水印提取

图片数字水印功能特性

类别	说明
被检原图片格式	png、jpg、bmp、tiff、webp
水印类型	支持图片、文本两种类型水印嵌入
抗攻击性	能够抵抗一定程度的画面裁剪、遮挡、缩放、截屏攻击

产品优势

高质量

通过AI模型深度学习，根据视频复杂度动态分配最优编码参数，基于人眼主观的画面增强，支持画质修复、超分辨率，并结合codec本身的深度调优，大幅提升画面质量。

高效率

根据用户级别、队列级别、视频时长和复杂度等综合因素智能调度，确保高优任务优先处理。长文件分片并行处理大幅提升转码速度。

强稳定

专属集群保障转码环境强稳定，分布式部署动态扩展，灵活应对业务量激增，转码异常实时监控报警，7*24h技术服务支持。

低成本

百度智能云转码服务全网价格最低，让您花最低的成本获得最优的服务。此外，智感超清转码能帮助节省大量带宽和存储成本。

高安全

百度智能云对象存储BOS，服务可用性99.9%，可靠性99.999999999%，具有多级安全控制和防护，全方位保障数据安全。

使用限制

产品限制	具体说明
服务区域	目前支持华北-北京、华南-广州、华东-苏州三个地域。 如果没有特定需求，请优先选择华北-北京。
任务队列	每个账号队列总量量为100。即每个账号最多可同时处理100个转码/截图任务。若想扩容可提交工单申请。
转码作业	单账号提交作业的最大速度不可超过100次每秒。 单账号查询作业的最大速度不可超过100次每秒。
极速转码	极速转码针对长视频，通过分片策略单文件并行转码，大大提高转码速度。极速转码目前为白名单开放，若需要使用请提交工单申请开通，申请时请提供userID和pipeline名称。
媒资存储	MCP转码输入和输出媒资文件均存储在BOS里，同一账号必须在BOS里拥有至少一个bucket。
感知编码	• 感知编码目前支持华北-北京、华东-苏州、华南-广州。 • 感知编码支持系统预置的模板，也同时支持自定义模板。若系统模板不能满足您的要求，可进行自定义模板。
智感超清	• 智感超清目前仅支持华北-北京区域。 • 智感超清系列暂不支持自定义模板，您可使用系统预置的模板（画质提升、老片修复），若系统模板不能满足您的要求，请提交工单申请新建模板。 • 画质提升内置了6个模板，包括：mcp.video*mp4_720p_h264_sr（将低于720p的视频超分到720p）、mcp.video_mp4_1080p_h264_sr（将低于1080p的视频超分到1080p）、mcp.video_mp4_1440p_h264_sr（将低于1440p的视频超分到1440p）、mcp.video_mp4_4k_h265_sr（将低于2160p的视频超分到2160p）、mcp.video_mp4_4k_h265_sr_hdr（将低于2160p的视频超分到2160p, 并输出hdr视频）、mcp.video_mp4_4k_h265_sr_hdr_cz（将低于2160p的视频超分到2160p, fps提升到60, 并输出hdr视频）。 • 老片修复内置了4个模板，包括：mcp.video_mp4_follow_scratch_noise_enhan（去划痕,去噪，画质增强，分辨率保持不变，支持1080p以下输入）、mcp.video_mp4_follow_scratch_noise_color（去划痕,去噪，黑白视频上色，分辨率保持不变，支持1080p以下输入）、mcp.video_mp4_follow_scratch_noise（去划痕,去噪，分辨率保持不变，支持1080p以下输入）、mcp.video_mp4_follow_scratch（去划痕，分辨率保持不变，支持1080p以下输入）。 • 智感超清目前为白名单开放，若需要使用请提交 <ahref="https://ticket.bce.baidu.com/=1556165872646/#/ticket/create~productId=34">工单申请开通，申请时请提供userID和pipeline名称。

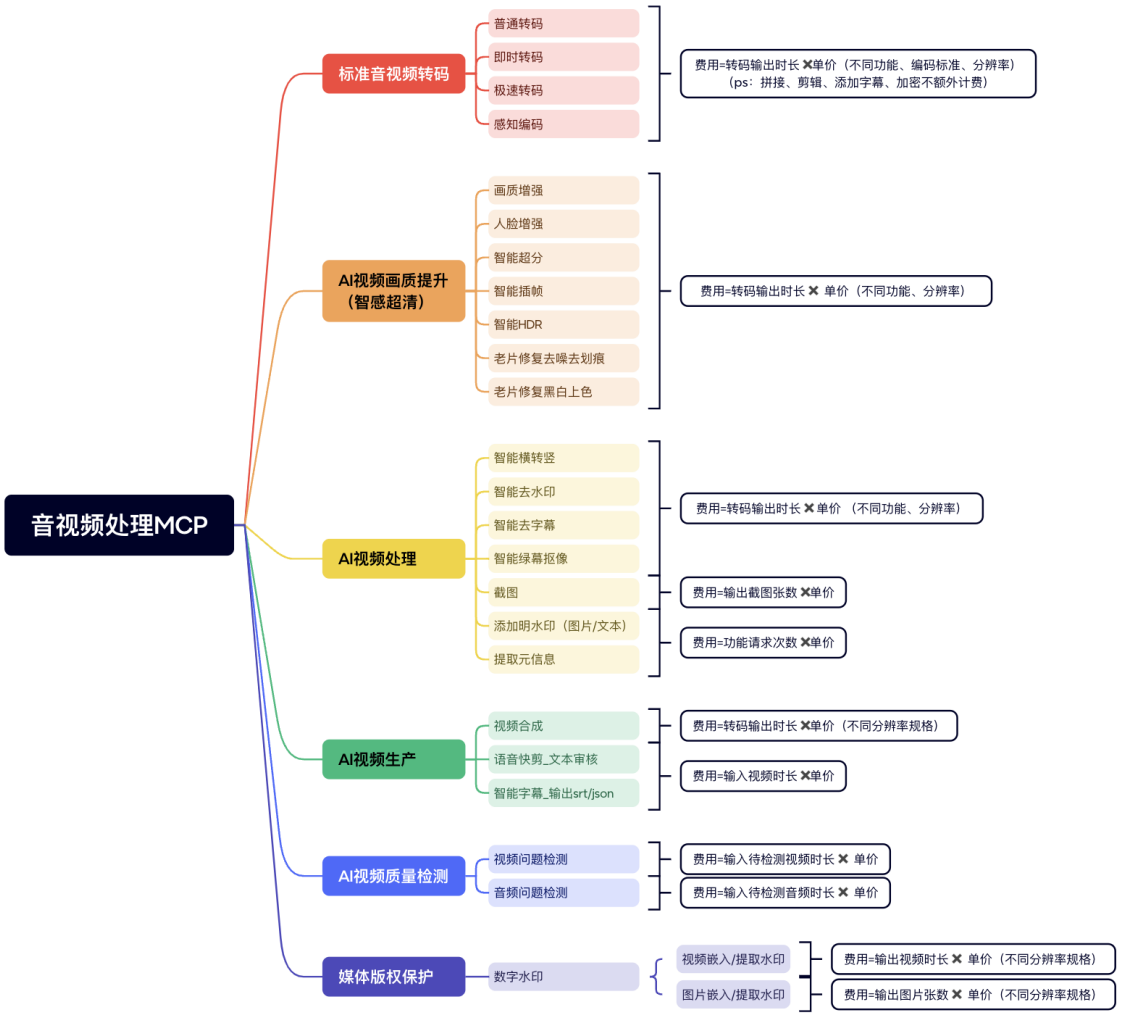
产品计费

计费概述

重要通知：

1. 自2024年3月27日起，本产品计费项组成和计费项名称进行更新，详情说明请参见下文；
2. 自2024年3月27日起，本产品对外正式计费的有：标准音视频转码、AI视频画质提升（智感超清）、AI视频处理、AI视频质量检测、媒体版权保护。

计费项组成



计费方式

音视频处理产品提供两种计费方式：预付费、后付费。

预付费

- 付费方式：预先购买混合时长转码包，从转码包剩余时长中抵扣。
- 生成账单时优先抵扣转码包，多个包时有效期先结束的优先扣减，当所有包扣减完毕后，自动转为按量计费的方式扣减。

后付费

根据您的使用情况，各计费项单独计量计费，不使用不计费。

计费和账单周期

付费方式	计费周期	账单周期	示例1
预付费	小时	按小时对您的使用费用进行出账并扣费，即对过去 60 分钟内的分钟级别账单进行加和，出具小时账单；出账单时间是当前计费周期结束后 1 小时左右，如 10:00 - 11:00 的账单会在 12:00 左右生成，具体以系统出账时间为准。	在10:00-11:00成功购买混合转码包1000分钟，则12:00之前生成流水账单：17元。
后付费	小时	按小时扣费，即北京时间整点扣费并生成账单；出账单时间是当前计费周期结束后1小时内。例如，10:00-11:00的账单会在12:00之前生成，具体以系统出账时间为准。	在10:00-11:00成功输出H265 1920*1080规格的视频1分钟，则12:00之前生成流水账单：1*0.3092=0.3092元；具体以系统出账时间为准。

定价详情

百度云音视频处理各计费项定价，请参见音视频处理MCP价格说明。

其他计费项

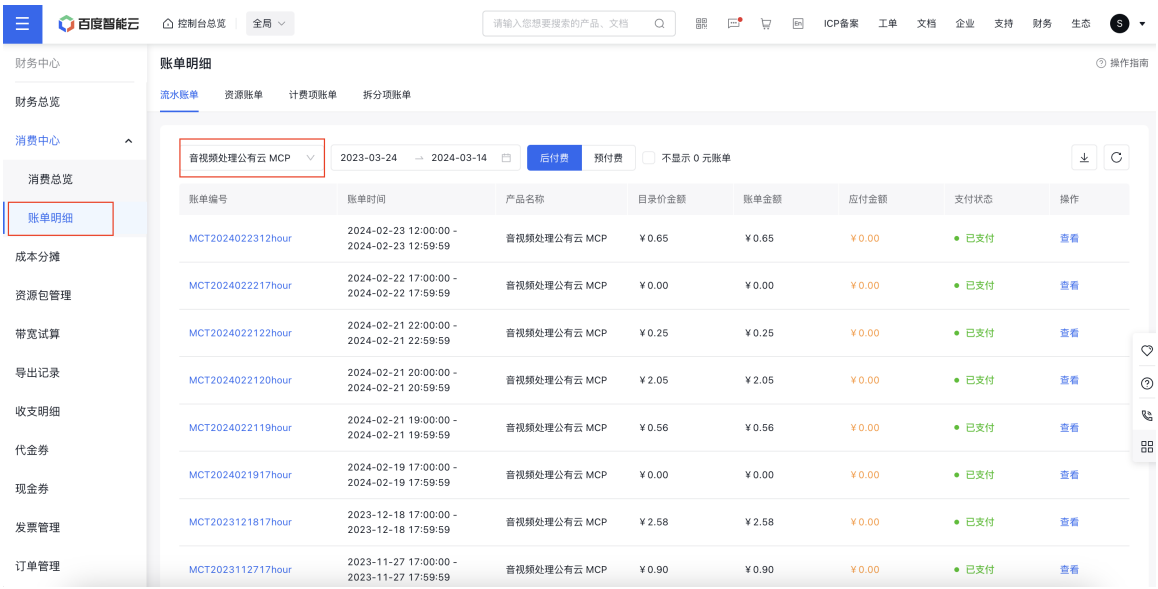
产品	计费项	相关文档
对象存储	存储及流量费用	BOS计费说明
（可选）内容分发网络 CDN	CDN 加速服务	CDN计费说明

账单查询

操作步骤

1. 登录百度智能云；
2. 在右上角菜单栏选择财务：

3. 点击左侧栏消费中心 > 账单明细 ；
4. 默认展示流水账单，选择产品名称：音视频处理公有云 MCP，默认选择后付费：



计费项说明

音视频转码计费

定价

标准转码

转码输出规格	单价（元/分钟）
H.264 4K（3840*2160）及以下	0.2660
H.264 2K（2560*1440）及以下	0.1330
H.264 HD（1920*1080）及以下	0.0618
H.264 SD（1280*720）及以下	0.0310
H.264 LD（640*480）及以下	0.0152
H.265 4K（3840*2160）及以下	1.3300
H.265 2K（2560*1440）及以下	0.6650
H.265 HD（1920*1080）及以下	0.3092
H.265 SD（1280*720）及以下	0.1547
H.265 LD（640*480）及以下	0.0760
AV1 4K（3840*2160）及以下	2.3940
AV1 2K（2560*1440）及以下	1.1970
AV1 HD（1920*1080）及以下	0.5566
AV1 SD（1280*720）及以下	0.2787
AV1 LD（640*480）及以下	0.1520
视频转封装	0.0067
纯音频转码	0.0053

感知编码

转码输出规格	单价（元/分钟）
BD.264 4K（3840*2160）及以下	0.7980
BD.264 2K（2560*1440）及以下	0.3990
BD.264 HD（1920*1080）及以下	0.1855
BD.264 SD（1280*720）及以下	0.0929
BD.264 LD（640*480）及以下	0.0618
BD.265 4K（3840*2160）及以下	3.9900
BD.265 2K（2560*1440）及以下	1.9950
BD.265 HD（1920*1080）及以下	0.9277
BD.265 SD（1280*720）及以下	0.4640
BD.265 LD（640*480）及以下	0.3092
BD.AV1 4K（3840*2160）及以下	7.9800
BD.AV1 2K（2560*1440）及以下	3.9900
BD.AV1 HD（1920*1080）及以下	1.8582
BD.AV1 SD（1280*720）及以下	0.9291
BD.AV1 LD（640*480）及以下	0.6213

即时转码

转码输出规格	单价（元/分钟）
H.264 HD（1920*1080）及以下	0.1237
H.264 SD（1280*720）及以下	0.0619
H.264 LD（640*480）及以下	0.0412

极速转码

转码输出规格	单价（元/分钟）
H.264 4K（3840*2160）及以下	0.5320
H.264 2K（2560*1440）及以下	0.2660
H.264 HD（1920*1080）及以下	0.1237
H.264 SD（1280*720）及以下	0.0619
H.264 LD（640*480）及以下	0.0412
H.265 4K（3840*2160）及以下	2.6600
H.265 2K（2560*1440）及以下	1.3300
H.265 HD（1920*1080）及以下	0.6185
H.265 SD（1280*720）及以下	0.3093
H.265 LD（640*480）及以下	0.2062

计费规则

- 按不同格式的转码时长进行计费，费用计算公式：转码费用 = 输出文件时长 x 不同规格的转码单价
- 付费方式：后付费
- 适用地域：北京、广州、苏州
- 输出时长：对于每个转码输出文件，按分钟计费，文件时长精确到小数点后两位，第二位根据第三位四舍五入。
- 转码规格：输出规格按输出视频分辨率的长边和短边属于输出规格划定的范围进行判定，方式如下：以输出SD (1280 x 720) 规格为例，输出视频的分辨率长边不大于1280且短边不大于720则属于该输出规格。如输出视频的长边大于1280或输出视频的短边大于720，该输出视频属于更高输出规格。
- 详细价格信息，请参见“音视频处理价格说明”。

计费示例

在10:00-11:00成功输出标准转码H264 1920✖1080规格的视频1分钟、H265 1920✖1080规格的视频2分钟，则12:00之前生成流水账单：1✖0.0618+2✖0.3092=0.6802元；具体以系统出账时间为准。

智感超清计费项

定价

画质增强（细节增强/色彩增强）

计费项	单价（元/分钟）
画质增强_4k（3840*2160）及以下	1.5
画质增强_2k（2560*1440）及以下	0.7
画质增强_HD（1920*1080）及以下	0.35
画质增强_SD（1280*720）及以下	0.2
画质增强_LD（640*480）及以下	0.15

ps：当用户调用细节增强/色彩增强功能时，账单内显示“画质增强”的计费项名称。

人脸增强

计费项	单价（元/分钟）
人脸增强	3.0

智能超分

计费项	单价（元/分钟）
智能超分_4k（3840*2160）及以下	3.6
智能超分_2k（2560*1440）及以下	1.6
智能超分_HD（1920*1080）及以下	0.9
智能超分_SD（1280*720）及以下	0.5

智能HDR

计费项	单价（元/分钟）
智能HDR_4k（3840*2160）及以下	2.1
智能HDR_2k（2560*1440）及以下	1.2
智能HDR_HD（1920*1080）及以下	0.6
智能HDR_SD（1280*720）及以下	0.5
智能HDR_LD（640*480）及以下	0.5

智能插帧

计费项	单价（元/分钟）
智能插帧_4k（3840*2160）及以下	5.8
智能插帧_2k（2560*1440）及以下	2.8
智能插帧_HD（1920*1080）及以下	1.5
智能插帧_SD（1280*720）及以下	0.7
智能插帧_LD（640*480）及以下	0.3

老片修复

计费项	单价（元/分钟）
老片修复_去噪/去划痕_HD（1920*1080）及以下	6.0
老片修复_去噪/去划痕_SD（1280*720）及以下	3.0
老片修复_去噪/去划痕_LD（640*480）及以下	2.0
老片修复_黑白上色_HD（1920*1080）及以下	0.8
老片修复_黑白上色_SD（1280*720）及以下	0.8
老片修复_黑白上色_LD（640*480）及以下	0.7

计费规则

- 按不同格式的转码时长进行计费，费用计算公式：转码费用 = 输出文件时长 x 不同规格的转码单价
- 付费方式：后付费
- 适用地域：北京、广州、苏州
- 输出时长：对于每个转码输出文件，按分钟计费，文件时长精确到小数点后两位，第二位根据第三位四舍五入。
- 转码规格：输出规格按输出视频分辨率的长边和短边属于输出规格划定的范围进行判定，方式如下：以输出SD (1280 x 720) 规格为例，输出视频的分辨率长边不大于1280且短边不大于720则属于该输出规格。如输出视频的长边大于1280或输出视频的短边大于720，该输出视频属于更高输出规格。
- 详细价格信息，请参见“音视频处理价格说明”。

计费示例

示例1:在10:00-11:00同时开启色彩增强、细节增强，并输出分辨率1920✖1080规格的视频10分钟，则12:00之前生成流水账单：10✖0.35=3.5元；具体以系统出账时间为准。
示例2:在10:00-11:00同时开启去噪、去划痕，并输出分辨率1920✖1080规格的视频10分钟，则12:00之前生成流水账单：10✖6.0=60元；具体以系统出账时间为准。

AI视频处理与生产计费项

定价

智能横转竖

计费项	单价（元/分钟）
智能横转竖_4k（3840*2160）及以下	1.5
智能横转竖_2k（2560*1440）及以下	0.3
智能横转竖_ HD（1920*1080）及以下	0.2
智能横转竖_ SD（1280*720）及以下	0.15
智能横转竖_ LD（640*480）及以下	0.08

智能去水印

计费项	单价（元/分钟）
智能去水印_4k（3840*2160）及以下	1.5
智能去水印_2k（2560*1440）及以下	0.3
智能去水印_HD（1920*1080）及以下	0.2
智能去水印_SD（1280*720）及以下	0.15
智能去水印_LD（640*480）及以下	0.08

智能去字幕

计费项	单价（元/分钟）
智能去字幕_4k（3840*2160）及以下	2.8
智能去字幕_2k（2560*1440）及以下	0.8
智能去字幕_HD（1920*1080）及以下	0.7
智能去字幕_SD（1280*720）及以下	0.4
智能去字幕_LD（640*480）及以下	0.2

智能绿幕抠像

计费项	单价（元/分钟）
智能绿幕抠像_4k（3840*2160）及以下	4.8
智能绿幕抠像_2k（2560*1440）及以下	1.2
智能绿幕抠像_HD（1920*1080）及以下	1.1
智能绿幕抠像_SD（1280*720）及以下	0.5
智能绿幕抠像_LD（640*480）及以下	0.25

计费规则

- 按不同输出分辨率时长进行计费，费用计算公式：费用 = 输出文件时长 x 不同规格的转码单价
- 付费方式：后付费
- 适用地域：北京、广州、苏州
- 输出规格：输出规格按输出视频分辨率的长边和短边属于输出规格划定的范围进行判定，方式如下：以输出SD (1280 x 720) 规格为例，输出视频的分辨率长边不大于1280且短边不大于720则属于该输出规格。如输出视频的长边大于1280或输出视频的短边大于720，该输出视频属于更高输出规格。
- 详细价格信息，请参见“音视频处理价格说明”。

计费示例

在10:00-11:00成功调用智能去字幕并输出规格为1920*1080的视频1分钟，成功调用智能绿幕抠像并输出规格为1280*720的视频5分钟，则12:00之前生成流水账单：1✖0.7+5✖0.5=1.2元；具体以系统出账时间为准。

AI视频质量检测计费项

定价

计费项	单价（元/分钟）
视频问题检测	0.08
音频问题检测	0.03

计费规则

- 费用计算公式：费用 = 输入待检测文件时长 x 单价
- 付费方式：后付费
- 适用地域：北京、广州、苏州
- 输入规格：不区分输入视频规格，统一价格。
- 详细价格信息，请参见“音视频处理价格说明”。

计费示例

在10:00-11:00输入视频1分钟成功发起一个视频问题检测任务，输入视频2分钟成功发起一个音频问题检测任务，则12:00之前生成流水账单：1✖0.08+2✖0.03=0.14元；具体以系统出账时间为准。

媒体版权保护计费项

定价

视频嵌入/提取数字水印

计费项	单价（元/分钟）
4K（3840*2160）	0.5
2K（2560*1440）	0.45
HD（1920*1080）	0.38
SD（1280*720）	0.25
LD（640*480）	0.15
提取数字水印	0.5

图片嵌入/提取数字水印

计费项	单价（元/张）
添加数字水印	0.001
提取数字水印	0.001

计费规则

- 费用计算公式1：视频添加/提取数字水印费用 = 输出文件时长 x 不同规格的单价
- 费用计算公式2：图片添加/提取数字水印费用 = 调用添加/提取数字水印次数 x 单价
- 付费方式：后付费
- 适用地域：北京、广州、苏州
- 输出规格：输出规格按输出视频分辨率的长边和短边属于输出规格划定的范围进行判定，方式如下：以输出SD (1280 x

720) 规格为例，输出视频的分辨率长边不大于1280且短边不大于720则属于该输出规格。如输出视频的长边大于1280或输出视频的短边大于720，该输出视频属于更高输出规格。

- 详细价格信息，请参见“音视频处理价格说明”。

计费示例

场景1：在10:00-11:00成功嵌入数字水印输出规格为1440*1080的视频 1分钟， 成功提取数字水印并输出规格为1280✖720的视频 2分钟，则12:00之前生成流水账单：1✖0.38+2✖0.001=0.381元；具体以系统出账时间为准。
场景2：在10:00-11:00成功调用提取数字水印任务并输出图片9次，除此之外无调用其他计费项，总费用为9✖0.001=0.009元（少于0.01元），但由于账单总费用少于0.01元，因此12:00之前不生成流水账单；紧接着，在11:00-12:00又成功调用提取数字水印任务并输出图片1次，则13:00之前生成流水账单：9✖0.001+1✖0.001=0.01元。

预付费资源包

定价

转码包规格（分钟）	单价（元）	折扣
1000	17	0.80
1000	17	0.80
5,000	77	0.75
20,000	280	0.68
100,000	1195	0.58
500,000	5150	0.50
1,000,000	8240	0.40
2,000,000	12360	0.30

您可直接在MCP管理控制台的“转码包”菜单页面，点击【购买转码包】直接购买。

抵扣规则

编码规格	折扣时长比例
H.264 4K（3840*2160）及以下	13:1
H.264 2K（2560*1440）及以下	6.5:1
H.264 HD（1920*1080）及以下	3:1
H.264 SD（1280*720）及以下	1.5:1
H.264 LD（640*480）及以下	1:1
H.265 4K（3840*2160）及以下	65:1
H.265 2K（2560*1440）及以下	32.5:1
H.265 HD（1920*1080）及以下	15:1
H.265 SD（1280*720）及以下	7.5:1
H.265 LD（640*480）及以下	5:1

(抵扣时长比例=抵扣转码包时长：实际转码输出时长 ）

计费规则

- 费用计算公式：费用 =下单数量 x 单价

- 付费方式：预付费
- 适用地域：北京、广州、苏州
- 使用范围： **当前此转码包仅适用于标准转码H264 LD、H264 SD、H264 HD、H264 2K、H264 4K、H265 LD、H265 SD、H265 HD、H265 2K、H265 4K十种计费项扣费（不包含极速转码、感知编码、AI画质提升等其他计费项）。**
- 详细价格信息，请参见“音视频处理价格说明”。

计费示例

假设您在10:00-11:00成功下单购买1个规格为1000000分钟的转码包，则12:00之前生成流水账单：1x8240=8240元，具体以系统出账时间为准。若在任意时间内成功输出普通转码H264 HD规格的视频1分钟，则转码包剩余用量为：1000000-3x1=999997分钟。

特殊问题说明

退款问题

预付费退款规则

官方退款：

- 若用户购买时长包7天内且未使用，可前往退订管理页面进行自助退订。
- 若已使用或超出7天，则不支持退款。

特殊退款：

- 不符合上述退款条件、仍需退款的用户，若时长包有效且有剩余流量，需提交工单走销售审批，人工osp操作进行退款。退款金额=（已抵扣时长/转码包时长）× 时长包购买金额。
- 资源包到期后，剩余流量不支持退订和退款。

余额不足和欠费说明

实例类型：后付费

实例状态

余额不足

说明：根据您最近3天的账单金额来判断您的账户余额（含可用代金券）是否足够支付未来3天的费用，若不足以支付，系统发送续费提醒。
处理方法：给百度智能云账号充值。

欠费处理

说明：

- 北京时间整点检查您的账户余额是否足以支付本次MCP账单的费用（如北京时间11点整检查账户余额是否足以支付10点至11点的账单费用），若不足以支付，即为欠费，欠费时系统会发送欠费通知。
- 欠费后百度智能云将停止为您服务，即您不能再提交任何新的作业，但已提交的作业会继续执行直至结束。您在欠费之日15天内（至暂停之时起至360个小时满（15 x 24）为15天届满）充值补足欠费后，被暂停的服务将恢复，您可继续使用百度智能云的MCP音视频处理服务；欠费超过15天（至暂停之时起至360个小时满（15X24）为15天届满），不再保证您的数据完整性，可能会随时被清理。

处理方法：给百度智能云账号充值，服务立即恢复正常。

快速入门

MCP快速使用流程

本文介绍如何使用控制台快速使用音视频处理MCP。

使用控制台进行音视频处理的基本操作流程如下图所示：

1. 开通服务：开通音视频处理MCP服务。
2. 上传文件至BOS：音视频转码服务输入输出依赖于对象存储服务（BOS），需要先将文件上传至BOS。
3. 创建任务队列：创建任务队列管理转码任务。
4. 创建转码任务：创建音视频转码任务。

以上流程适用于控制台操作，如果您是API或SDK用户，请参见API参考、SDK参考。

开通服务

前提条件

使用百度智能云音视频处理MCP前，您需要完成以下操作：

- 注册并登录百度智能云平台，请参考注册和登录。
- 完成实名认证，操作细节请参考实名认证，实名认证后才可购买百度智能云音视频处理MCP。

开通服务

1. 登录管理控制台，在控制台导航栏中选择“产品服务 > 智能视频 > 音视频处理MCP”。
2. 在开通音视频处理MCP页面，点击 **立即开通**，即可开通音视频处理MCP服务，同时默认开通BOS服务。

上传文件到BOS

概述

音视频转码服务输入输出依赖百度智能云对象存储服务（BOS），即将存储于BOS上的源音视频文件转码为目标音视频文件并存储在BOS上。本文档将以BOS控制台为例创建一个BOS bucket，并将源音视频文件上传至该bucket。

上传文件到BOS

1. 登录 BOS管理控制台 ，在控制台左侧的导航栏中找到并点击 + 按钮（“新建Bucket”按钮），在弹出框中按照提示创建Bucket。

2. 在 **存储管理** 区域选择创建的Bucket，在右侧文件列表页面，点击 **上传文件** 按钮。

3. 您可以使用拖拽或直接选择文件的方式上传文件。

相关API及SDK

- 使用 API 上传 Object：
  - PutObject 接口
- 使用 SDK 上传 Object ：
  - Java SDK
  - Python SDK
  - PHP SDK

创建任务队列

概述

任务队列是用来管理转码任务的，创建任务队列时，需指定转码任务的输入和输出bos bucket。本文档将以控制台为例创建任务队列。

创建任务队列

1. 登录 音视频处理控制台。
2. 在左侧导航选择 **产品配置** > **任务队列**，进入任务队列页面。
3. 点击 **创建队列**，进入 **新建队列**页面。

4. 填写相关信息，点击 **创建队列**，完成队列的创建。

5. 创建完成后，您可以在队列列表查看已创建队列的概要信息。

相关API

- 使用API创建任务队列：
  - 创建队列接口
- 使用SDK创建任务队列：
  - Java SDK
  - PHP SDK

创建转码任务

（可选）创建转码模板

转码模板是转码具体参数的配置集合，目前支持自定义模板及系统内置的转码模板。如果系统提供的转码模板不能很好地满足转码需求，您可以通过控制台、API、SDK自定义转码模板。

操作步骤

1. 登录 音视频处理控制台。
2. 在左侧导航选择 **产品配置** > **转码模板**，进入 **转码模板**页面。
3. 点击 **新建转码模板**，进入 **新建转码模板** 页面。

4. 您可以在空白模板上填写相应内容，也可以选择一个系统模板，在此基础上进行修改，以完成自定义转码模板的创建。

创建转码任务

转码任务即完成音视频转码操作的一个任务。您需要给每个转码任务选择一个任务队列，提供源音视频文件所在的BOS路径和目标音视频文件名称，同时选择一个转码模板完成特定规格的转码需求。

操作步骤

1. 登录 音视频处理控制台。

2. 在左侧导航选择 **任务管理 > 转码任务**，在任务队列列表点击 **新建转码任务**。

3. 在 **创建转码任务** 页面，输入相应的信息。

4. 输入完成后，点击**立即创建**，完成转码任务创建。

5. 进入创建任务列表，如果转码任务状态变更为 **成功**，说明转码成功。

（可选）CDN视频加速

转码完成后，您还可以通过百度CDN 加速服务将存储在BOS bucket上的目标音视频文件进行安 全加速，实现目标音视频文件快速分发。 您可通过控制台对目标音视频所在的BOS bucket 进行全网加速，具体请参考CDN加速发布。

相关API

创建转码模板

- 使用API创建模板：
  - 创建模板接口
- 使用SDK创建模板：
  - Java SDK
  - PHP SDK

创建转码任务

- 使用API创建转码任务：
  - 创建转码任务接口
- 使用SDK创建转码任务：
  - Java SDK
  - PHP SDK

创建即时转码任务

基本介绍

MCP提供了业界全新的视频即时转码接口，支持通过设置url参数的方法实时转换视频的码率，实现h264，h265两种码率的互转。

操作方法

注意事项

- 使用前需开通百度智能云音视频处理服务（MCP）；
- 建议的转码视频分辨率：不超过1080P；
- 建议的转码视频大小：小于1GB。

规则和限制

- 目前仅支持北京、苏州、广州使用。

操作指南

概述

介绍

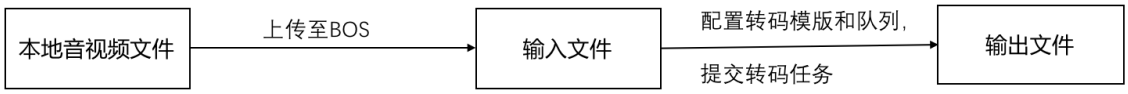
音视频转码主要提供对音视频的转码服务。其中输入输出依赖于百度智能云的对象存储（BOS），CDN加速分发依赖于百度智能云的CDN服务。转码后的视频播放遵循标准规范，可以使用通用的播放器，也可以使用百度智能云提供的播放器。

音视频转码预制了一部分系统内置转码模板，用户转码时需要的参数配置，可以通过下拉列表进行选择，方便了用户进行转码。如果系统模板不能满足您的需求，可以通过操作控制台console或者API方式自定义参数转码，以满足您的需求。具体请参考API参考。

前提条件

- 需要注册百度智能云账号，启用对象存储服务（创建bucket，获取访问权限）
- 需要完成实名认证，否则用户无法创建Bucket。
- （可选）申请CDN服务。

操作流程



1. 使用BOS的控制台/API/SDK将源音视频文件上传至BOS，作为转码任务的一个输入。
2. 根据目标输出音视频文件的规格需求，选择转码模板。
3. 提交转码任务到任务队列，转码输出。
4. 输出结果存储在用户所选择的任务队列对应的输出Bucket中。

视频上传与存储

概述

在使用视频处理服务MCP之前，您至少需要拥有一个Bucket（在对象存储BOS产品中创建）。建议您为每个队列设置两个Bucket，一个存储源视频，一个存储转码后的视频。对象存储BOS提供了便捷的文件存储和管理功能包括Bucket创建、文件上传等。

视频上传与存储

1. 登录BOS管理控制台，在控制台左侧的导航栏中找到并点击 + 按钮（“新建Bucket”按钮），在弹出框中按照提示创建Bucket。

2. 点击 **确定**，完成Bucket的创建。关于Bucket创建时，配置项的详细说明，请参见创建Bucket。

3. 在 **存储管理** 区域选择创建的Bucket，在右侧文件列表页面，点击 **上传文件** 按钮。

4. 您可以使用拖拽或直接选择文件的方式上传文件。关于上传文件的详细说明，请参见上传文件。

5. 更多的BOS产品信息和Bucket使用方式，请参见BOS的操作指南。

相关API及SDK

创建Bucket

- 通过API创建Bucket：
  - PutBucket 接口

- 通过SDK创建Bucket：
  - Java SDK
  - Python SDK
  - PHP SDK

上传Object

- 使用API上传Object：
  - PutObject 接口
- 使用SDK上传Object ：
  - Java SDK
  - Python SDK
  - PHP SDK

队列管理

创建队列

概述

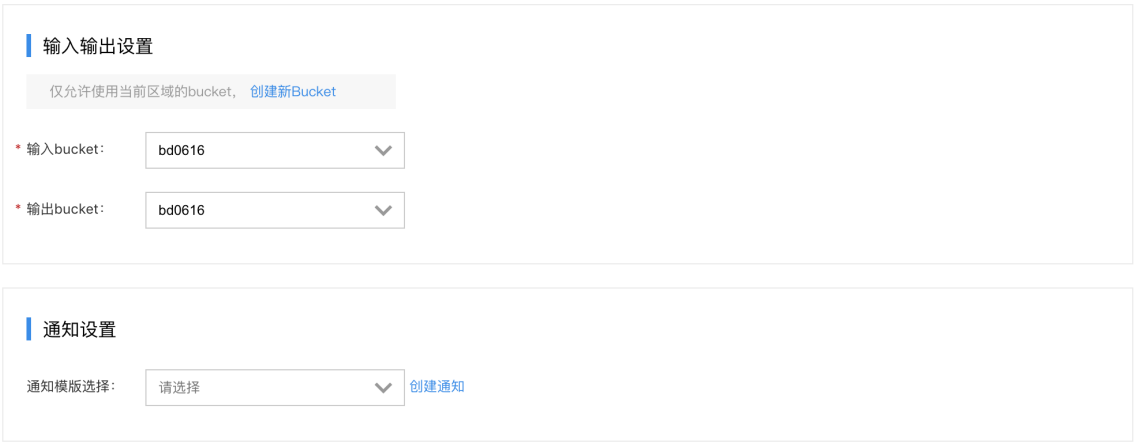
通过队列，您可以更灵活地管理转码任务。当创建一个任务时，您必须为该任务指定所属的队列。您可通过本文档了解如何使用控制台创建队列。

注意事项

队列容量为队列中可以并行进行的转码任务数。每个账号的队列总容量为 100。您可以创建多个队列分配队列容量。当队列总量不满足业务需求时，可以通过工单申请扩容。

创建队列

1. 登录 音视频处理 MCP 控制台。
2. 在左侧导航选择 **产品配置** > **任务队列**，进入任务队列页面。
3. 点击 **创建队列**，进入 **新建队列**页面。



配置项	说明
队列类型	普通队列和倍速队列。普通队列为普通转码任务的队列；倍速队列针对长视频，通过分片策略并行转码，大大提高转码速度。
队列名称	队列名称由数字和字母组成，大小写不敏感，最长40个字符。
队列容量	指您可以并行进行的转码任务数。队列容量需要小于等于剩余队列总容量。
输入/输出bucket	设置您的音视频输入/输出Bucket。输入bucket和输出bucket可以是同一个bucket。当您的BOS中不存在bucket时，需要在BOS中新建一个bucket。
通知模板	（选填）可以选择一个通知回调地址，当队列中的任务处理完成时，通过通知回调地址通知您。

5. 完成队列信息配置后，点击 **创建队列** 完成队列创建。
6. 创建完成后，您可以在队列列表查看队列的概要信息。

相关API及SDK

- 使用API创建队列：
  - 创建队列接口
- 使用SDK创建队列：
  - Java SDK
  - PHP SDK

队列编辑

概述

在队列编辑页面，除了队列名称和队列类型不可编辑外，其他信息均可编辑。

队列编辑

1. 登录 音视频处理 MCP 控制台。
2. 在左侧导航选择 **产品配置** > **任务队列**，进入任务队列页面。
3. 选择需要编辑的队列，在右侧操作栏中点击 **编辑**，打开更新队列页面。
4. 根据您的需求对队列信息进行编辑，您可编辑除 **队列类型**、**队列名称** 以外的信息。

配置项	说明
队列容量	指您可以并行进行的转码任务数。队列容量需要小于等于剩余队列总容量。
输入/输出bucket	设置您的音视频输入/输出Bucket。输入bucket和输出bucket可以是同一个bucket。当您的BOS中不存在bucket时，需要在BOS中新建一个bucket。
通知模板	（选填）可以选择一个通知回调地址，当队列中的任务处理完成时，通过通知回调地址通知您。

5. 编辑完成，点击 **更新队列**，完成对队列的编辑。

相关API及SDK

- 使用API更新指定队列：
  - 更新指定队列接口
- 使用SDK更新指定队列：
  - Java SDK
  - PHP SDK

查看队列详情

概述

在队列详情页面，您可以查看队列信息和队列中的任务信息。

1. 任务信息：排队中的任务数、处理中任务数、成功的任务数、失败的任务数。
2. 基本信息：队列名称、队列容量、队列描述、创建时间、更新时间。
3. 配置信息：输入输出bucket、通知回调模板。

查看队列详情

1. 登录 音视频处理 MCP 控制台。
2. 在左侧导航选择 **产品配置** > **任务队列**，进入任务队列页面。
3. 选择 **队列名称** 下面的链接或者右侧操作栏的 **查看详情**，打开任务详情页面。

相关API及SDK

- 使用API查询指定队列：
  - 查询指定队列接口
- 使用SDK查询指定队列：
  - Java SDK
  - PHP SDK

删除队列

概述

在队列详情页，可以删除队列。

> **注意：删除队列后将不能恢复，正在处理任务的队列不允许删除。**

删除队列

1. 登录音视频管理 MCP 控制台。
2. 在左侧导航栏选择 **产品配置** > **任务队列** ，进入任务队列列表。
3. 点击目标队列操作栏下的 **删除** 按钮，弹出确认删除对话框。

4. 点击 **确定**，完成队列的删除。

相关API及SDK

- 使用API删除指定队列：
  - 删除指定队列
- 使用SDK删除指定队列：
  - Java SDK
  - Python SDK

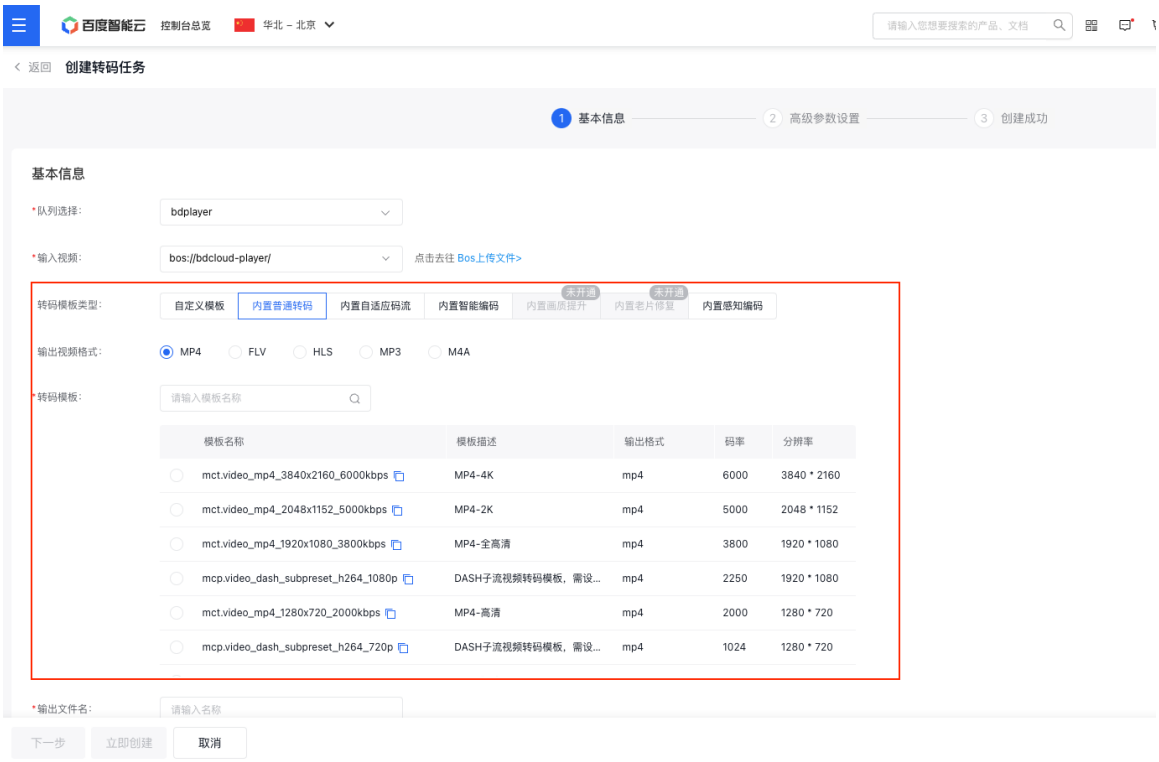
模板管理

转码模板

创建转码模板

概述

如果系统预设的Preset无法满足用户的需求，用户可以自定义自己的Preset。根据不同的转码需求，可以使用不同的接口创建Preset。查询方式可按照下方示意图：创建转码任务-选择转码模板类型、输出视频格式，或直接查找模板名称关键词定位到内置转码模板（内置画质提升/内置老片修复模板可通过提交工单开通查看/使用权限）。



系统内置模板

内置普通转码模板：

模板名称	模板描述
mct.video_mp4_3840x2160_6000kbps	MP4-4K
mct.video_mp4_2048x1152_5000kbps	MP4-2K
mct.video_mp4_1920x1080_3800kbps	MP4-全高清
mct.video_mp4_1280x720_2000kbps	MP4-高清
mct.video_mp4_854x480_1000kbps	MP4-标清
mct.video_mp4_640x360_600kbps	MP4-流畅
mct.video_mp4_320x180_300kbps	MP4-XLD
mct.video_mp4_transmux	转封装-MP4
mct.video_hls_3840x2160_6000kbps	hls-4K
mct.video_hls_2048x1152_5000kbps	hls-2K
mct.video_hls_1920x1080_3800kbps	hls-全高清
mct.video_hls_1280x720_2000kbps	hls-高清
mct.video_hls_848x477_1000kbps	hls-标清
mct.video_hls_640x360_600kbps	hls-流畅
mct.video_hls_320x180_300kbps	hls-XLD
mct.video_hls_transmux	转封装-hls
mct.audio_mp3_320kbps	Mp3-320
mct.audio_mp3_192kbps	MP3-192
mct.audio_mp3_160kbps	MP3-160
mct.audio_mp3_128kbps	MP3-128
mct.audio_mp3_64kbps	MP3-64

内置智感超清-感知编码模板：

模板名称	模板描述
mcp.video_mp4_4k_BD265_pvc	将低于4k的视频超分到4k，增强重点区域画面，并节省10%码率。
mcp.video_mp4_2k_BD265_pvc	将低于2k的视频超分到2k的mp4格式，增强重点区域画面，并节省10%码率。
mcp.video_mp4_1080p_BD265_pvc	将低于1080p的视频超分到1080p的mp4格式，增强重点区域画面，并节省10%码率。
mcp.video_mp4_480p_BD265_pvc	输出目标分辨率480p的mp4格式，增强重点区域画面，并节省10%码率。
mcp.video_mp4_360p_BD265_pvc	输出目标分辨率360p的mp4格式，增强重点区域画面，并节省10%码率。
mcp.video_mp4_720p_BD265_pvc	输出目标分辨率720p的mp4格式，增强重点区域画面，并节省10%码率。
mcp.video_flv_4k_BD265_pvc	将低于4k的视频超分到4k的flv格式，增强重点区域画面，并节省10%码率。
mcp.video_flv_2k_BD265_pvc	将低于2k的视频超分到2k的flv格式，增强重点区域画面，并节省10%码率。
mcp.video_flv_1080p_BD265_pvc	将低于1080p的视频超分到1080p的flv格式，增强重点区域画面，并节省10%码率。
mcp.video_flv_480p_BD265_pvc	输出目标分辨率480p的flv格式，增强重点区域画面，并节省10%码率。
mcp.video_flv_360p_BD265_pvc	输出目标分辨率360p的flv格式，增强重点区域画面，并节省10%码率。
mcp.video_flv_720p_BD265_pvc	输出目标分辨率720p的flv格式，增强重点区域画面，并节省10%码率。
mcp.video_hls_4k_BD265_pvc	将低于4k的视频超分到4k的hls格式，增强重点区域画面，并节省10%码率。
mcp.video_hls_2k_BD265_pvc	将低于2k的视频超分到2k的hls格式，增强重点区域画面，并节省10%码率。
mcp.video_hls_1080p_BD265_pvc	将低于1080p的视频超分到1080p的hls格式，增强重点区域画面，并节省10%码率。
mcp.video_hls_480p_BD265_pvc	输出目标分辨率480p的hls格式，增强重点区域画面，并节省10%码率。
mcp.video_hls_360p_BD265_pvc	输出目标分辨率360p的hls格式，增强重点区域画面，并节省10%码率。
mcp.video_hls_720p_BD265_pvc	输出目标分辨率720p的hls格式，增强重点区域画面，并节省10%码率。

内置智感超清-画质提升模板：

模板名称	模板描述
mcp.video_mp4_720p_h264_sr	将低于720p的视频超分到720p。
mcp.video_mp4_1080p_h264_sr	将低于1080p的视频超分到1080p。
mcp.video_mp4_1440p_h264_sr	将低于1440p的视频超分到1440p。
mcp.video_mp4_4k_h265_sr	将低于2160p的视频超分到2160p。
mcp.video_mp4_4k_h265_sr_hdr	将低于2160p的视频超分到2160p, 并输出hdr视频。
mcp.video_mp4_4k_h265_sr_hdr_cz	将低于2160p的视频超分到2160p, fps提升到60, 并输出hdr视频。

注意：对于超分任务，输入视频的宽高，都不能低于模板分辨率的1/3；画质提升功能需开通权限后使用，请根据控制台提示提交工单开通。

内置智感超清-老片修复模板：

模板名称	模板描述
mcp.video_mp4_follow_scratch_noise_enhan	去划痕,去噪，画质增强，分辨率保持不变，支持1080p以下输入。
mcp.video_mp4_follow_scratch_noise_color	去划痕,去噪，黑白视频上色，分辨率保持不变，支持1080p以下输入。
mcp.video_mp4_follow_scratch_noise	去划痕,去噪，分辨率保持不变，支持1080p以下输入。
mcp.video_mp4_follow_scratch	去划痕，分辨率保持不变，支持1080p以下输入。

注意：画质提升功能需开通权限后使用，请根据控制台提示提交工单开通。

内置自适应码流模板：

模板名称	模板描述
mcp.video_dash_subpreset_h264_1080p	自适应码流DASH转码模板输出分辨率1080P。
mcp.video_dash_separate_h264_720p	自适应码流DASH转码模板输出分辨率720P。
mcp.video_dash_subpreset_h264_360p	自适应码流DASH转码模板输出分辨率360P。

新建普通转码模板

1. 登录音视频处理 MCP 控制台。
2. 在左侧导航栏选择 **产品配置->转码模板**，进入转码模板页面。
3. 点击**新建音视频转码模板**，进入“新建转码模板”页面。

4. 用户根据需求填写相应配置信息：

参数	说明

模板名称	开头必须是小写字母，其余可以是小写字母或数字组成，最多不超过40个字符。
音/视频格式	指定音视频容器。其中，MP4为通用视频容器；FLV为Flash视频容器；HLS为流媒体视频容器；A-HLS是自适应码率HLS； MP3为MP3音频容器；M4A为MPEG4音频容器。
模板描述	（可选）输入模板描述。
样例模板	可以选择一个样例模板，参考已有参数创建模板。
视频参数	• 禁用视频：只从视频中提取音频时，可以开启“禁用视频”； • 编码标准：H.264是一种高压缩比的视频编码格式，能提供低码率、连续、流畅的高质量图像，而且具备很强的容错能力和网络适应能力。BD265为百度自研编码器； • 编码规格：指定目标视频所用编码规格，可以选择Baseline,Main或者High；面向中高端设备播放时(Web, iPhone4/Android3.0 以上)建议使用High/Main；低端嵌入式设备建议使用Baseline； • 码率控制方式：4种选择：CRF、VBR、CBR、CAE； • 最大码率：指定视频码率，单位kbps； 常用视频码率，低清：200 kbps；标清：600 kbps；高清：1200 kbps。若指定码率大于原视频码率，则使用原视频码率； • ROI增强：支持开/关。若开启，则编码器支持ROI人脸区域增强，默认关闭状态； • 恒定质量因子：若设置了crf值，最大码率表示最大目标码率； • 最高帧率：指定视频帧率，每秒显示的帧数，常用帧率为24，25，30等；Auto表示保持与输入一致；若指定帧率大于原视频帧率，则使用原视频帧率； • 最大尺寸：指定目标视频分辨率，默认保持与输入一致；若指定分辨率大于原视频分辨率，则使用原视频分辨率； • 伸缩策略：指定视频尺寸伸缩策略，可以选择Keep、ShrinkToFit和Stretch。Keep：保持输入尺寸比率；ShrinkToFit：自适应伸缩，根据需要加入黑边填充；Stretch，拉伸原视频尺寸； • 横竖版自适应：当原视频为竖形时，自动调整模板的宽小于高，保证缩放比最小，反之亦然；
音频参数	• 禁用音频：只从视频中提取视频时，可以开启“禁用音频”； • 编码标准：指定音频编码标准，AAC或者MP3,与音/视频容器关联； • 采样率：Auto表示保持与输入一致； • 码率：常用码率：40 kbps, 80 kbps, 128 kbps, 192 kbps等；若指定码率大于输入码率，则使用输入码率； • 声道：指定音频声道数，常用值为1或2；Auto表示保持与输入一致，若指定声道数大于输入声道数，则使用输入声道数； • 音频归一化：是否进行音频归一化操作； • 静音：可以选择音频静音。
高级参数	• 自动去片头黑帧：自动识别片头黑帧并去除； • 自动去黑边：自动识别黑边区域并去除黑边； • 自动去水印：自动识别水印并去除水印； • 剪辑开始时间：支持对视频进行时间维度剪辑，指定剪辑开始时间； • 持续时间：视频剪辑持续时长，默认全部视频；

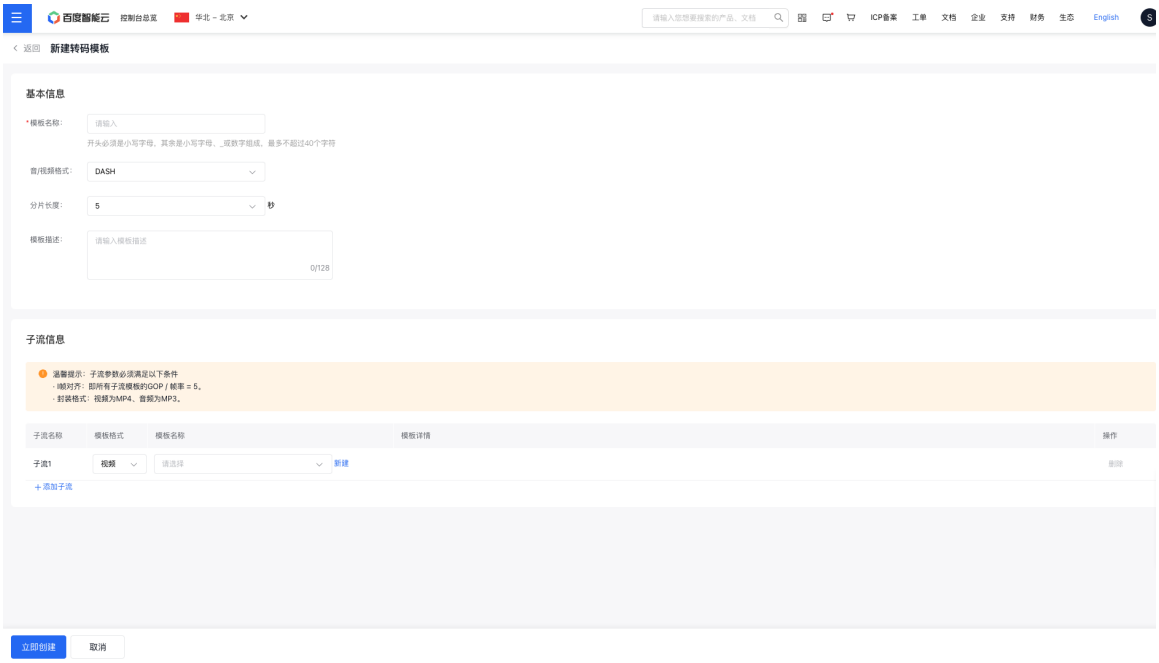
	• 持续时间：视频剪辑持续时长，默认全部视频； • 水印配置：视频添加水印； • 密钥策略：视频加密后，密钥获取策略，参考版权保护。

5. 点击**立即创建**，完成普通转码模板的创建。

新建自适应码流模板

1. 登录音视频处理 MCP 控制台。
2. 在左侧导航栏选择 **产品配置->转码模板** ,进入转码模板页面。
3. 点击**新建自适应码流模板**，进入“新建转码模板”页面。
4. 用户根据需求填写相应配置信息：

参数	说明
模板名称	开头必须是小写字母，其余可以是小写字母或数字组成，最多不超过40个字符。
音/视频格式	当前仅支持DASH。
分片长度	指规定输出分片最大时长（单位：秒）。
模板描述	（可选）输入模板描述。
子流信息配置	• 子流名称：按顺序默认命名子流名称； • 模板格式：支持子流输出2种转码封装格式：视频（.mp4）、音频（.mp3）； • 支持模板配置：选择已创建的模板，若无可用模板，请点击"新建"跳转至"新建音视频转码模板"页面进行创建普通转码模板； • 支持新增/删除子流，子流总数不超过10。



5. 点击**立即创建**，完成自适应码流模板的创建。

新建感知编码模板

1. 登录音视频处理 MCP 控制台。

2. 在左侧导航栏选择 **产品配置->转码模板** ,进入转码模板页面。

3. 点击**新建感知编码模板**，进入“新建转码模板”页面。

4. 用户根据需求填写相应配置信息：

参数	说明
模板名称	开头必须是小写字母，其余可以是小写字母或数字组成，最多不超过40个字符。
音/视频格式	当前仅支持MP4。
模板描述	（可选）输入模板描述。
样例模板	可以选择一个样例模板，参考已有参数创建模板。
视频参数	• 编码标准：提供2种百度自研编码器BD265/BD264选择； • 编码规格：指定目标视频所用编码规格，可以选择Baseline,Main或者High；面向中高端设备播放时(Web, iPhone4/Android3.0 以上)建议使用High/Main；低端嵌入式设备建议使用Baseline； • 最大码率：指定视频码率，单位kbps； 常用视频码率，低清：200 kbps；标清：600 kbps；高清：1200 kbps；若指定码率大于原视频码率，则使用原视频码率； • ROI增强：感知编码模板默认开启状态； • 最大尺寸：指定目标视频分辨率，默认保持与输入一致；若指定分辨率大于原视频分辨率，则使用原视频分辨率； • 最大I帧间隔GOP：指定I帧间隔距离，I帧间隔距离越大，P、B帧的数量就会越多,画面质量就会相对较好（若想创建dash子流模板，必须满足最大I帧间隔除以帧率等于5。）； • 伸缩策略：指定视频尺寸伸缩策略，可以选择Keep、ShrinkToFit和Stretch；Keep：保持输入尺寸比率；ShrinkToFit：自适应伸缩，根据需要加入黑边填充；Stretch，拉伸原视频尺寸。默认Keep策略； • 横竖版自适应：默认开启。当原视频为竖形时，自动调整模板的宽小于高，保证缩放比最小，反之亦然。
音频参数	（和普通转码模板配置相同。）
高级参数	（和普通转码模板配置相同。）

5. 点击**立即创建**，完成感知编码模板的创建。

相关API及SDK

- 使用API创建模板：
  - 创建模板
- 使用SDK创建模板：
  - Java SDK
  - Python SDK

查看转码模板

概述

创建转码模板后，用户可以在转码模板页面查看转码模板的详细信息。

查看转码模板

1. 登录音视频处理 MCP 控制台。
2. 在左侧导航栏选择 **产品配置->转码模板** ,进入转码模板页面。
3. 找到目标“模板名称”，点击其操作列的**查看详情**按钮。

4. 进入模板信息页面，在此页面可以查看**基本信息**、**视频参数**、**音频参数**、**高级参数**等信息。

编辑转码模板

概述

您可以根据不同的转码需求编辑转码模板，修改已创建的自定义转码模板。

编辑转码模板

1. 登录音视频处理 MCP 控制台。
2. 在左侧导航栏选择 **产品配置->转码模板** ，进入转码模板页面。
3. 找到目标“模板名称”，点击其操作列的**编辑**按钮。

4. 在更新转码模板页面，您可以修改以下配置信息：

5. 修改完成后点击**更新模板**，页面提示“转码模板更新成功”即更新模板成功。

相关 API

- 更新指定模板

删除转码模板

概述

如果您想要删除不需要的自定义转码模板，请按照以下步骤完成操作。

删除转码模板

1. 登录音视频处理 MCP 控制台。
2. 在左侧导航栏选择 **产品配置->转码模板** ，进入转码模板页面。
3. 找到目标“模板名称”，点击其操作列的**删除**按钮。

4. 在弹出的确认框中点击**确定，删除后将不能恢复，请谨慎操作**

水印模板

创建水印模板

概述

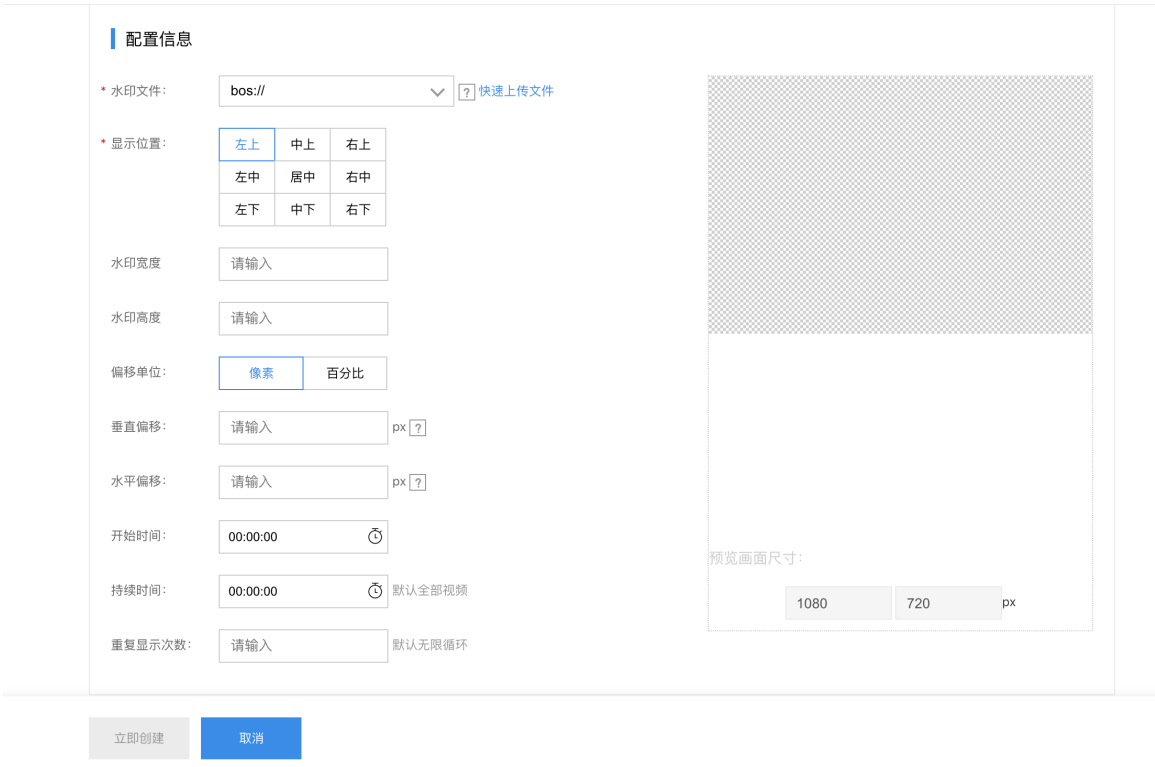
音视频处理可以自定义水印模板，您可通过本文档了解如何使用控制台创建水印模板。

前提条件

- 水印图标格式支持png（推荐）、jpg、gif格式。
- 关于如何将水印图片上传至BOS，请参考上传Object。

创建水印模板

1. 登录音视频处理 MCP 控制台。
2. 在左侧导航栏选择 **产品配置->水印模板** ,进入视频水印列表页面。
3. 点击**新建水印模板**，进入“新建水印模板”页面。



4. 用户根据需求填写相应配置信息：

- **显示位置**：水印图标在目标视频画面上的位置，采用九宫格定位策略。如下图所示，我们将目标视频画面等分为9个格子，然后根据图示每个格子默认位置，您可设置其水平偏移和垂直偏移。规则如下：
  - 偏移值为图片相对于九宫格每一个格边界的偏移量，偏移单位可以设置像素值和百分比值。
  - 图中为水平偏移和垂直偏移都为0时的默认位置。
  - 垂直偏移：该参数单位为像素时，取值范围0~3072；该参数单位为百分比时，取值范围0~100。
  - 水平偏移：该参数单位为像素，取值范围0~4096；该参数单位为百分比时，取值范围0~100。

- **水印宽度**、**水印高度**可以指定插入水印的大小。
- **偏移单位**：选择水印的定位方式。
- **开始时间**：指定水印出现的时间。
- **持续时间**：指定水印持续出现的时间，默认是全部视频。
- **重复显示次数**：指定水印重复出现的次数，默认是无限循环。

5. 点击**立即创建**，完成水印的创建。

说明：水印是创建自定义模板的可配置项。

> **举例**：当选择右上的时候，如果水平偏移填了5，垂直填了0，单位为像素值，表示相对于右上的九宫格距离右边界水平向左偏移了5个像素。

相关API及SDK

- 使用API创建水印：
  - 创建水印
- 使用SDK创建水印：
  - Java SDK
  - Python SDK

查看水印模板

概述

查看水印配置及预览效果。

查看水印模板

1. 登录音视频处理 MCP 控制台。
2. 在左侧导航栏选择 **产品配置->水印模板** ,进入视频水印列表页面。
3. 在视频水印列表页面点击目标水印模板右上角的  按钮，进入“水印详情”页面。
4. 用户根据需要查看水印基本信息及效果预览。

相关API及SDK

- 使用API水印接口：
  - 查询指定水印
  - 查询当前用户水印
- 使用SDK查看水印：
  - Java SDK
  - Python SDK

删除水印

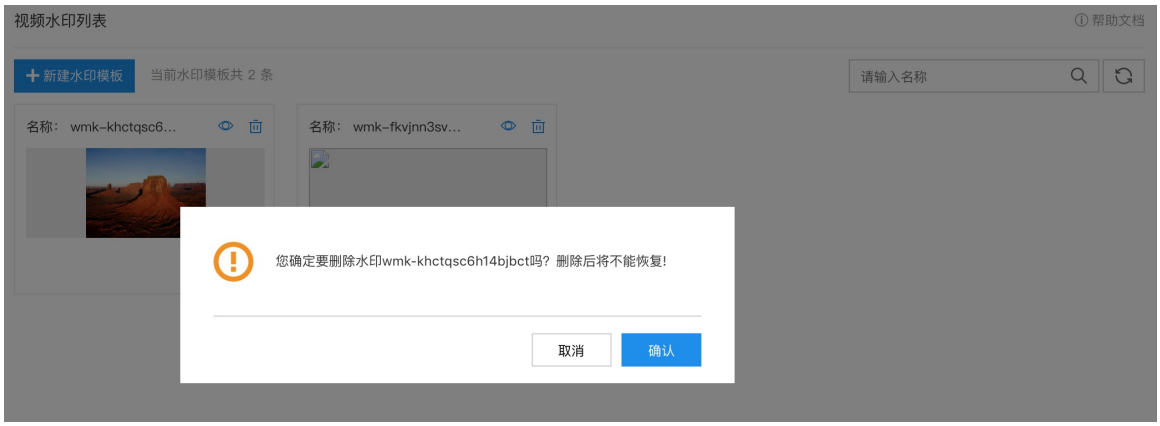
概述

用户不需要该水印时，将水印删除。

删除水印

> **注意**：删除后将不能恢复！请谨慎操作！

1. 登录音视频处理 MCP 控制台。
2. 在左侧导航栏选择 **产品配置->水印模板** ,进入视频水印列表页面。
3. 点击目标水印模板右上角的  按钮，弹出对话框。



4.点击 **确定**，完成水印的删除，删除的水印将不能恢复。

相关API及SDK

- 使用API删除水印：
  - 删除水印
- 使用SDK删除水印：
  - Java SDK
  - Python SDK

抽帧模板

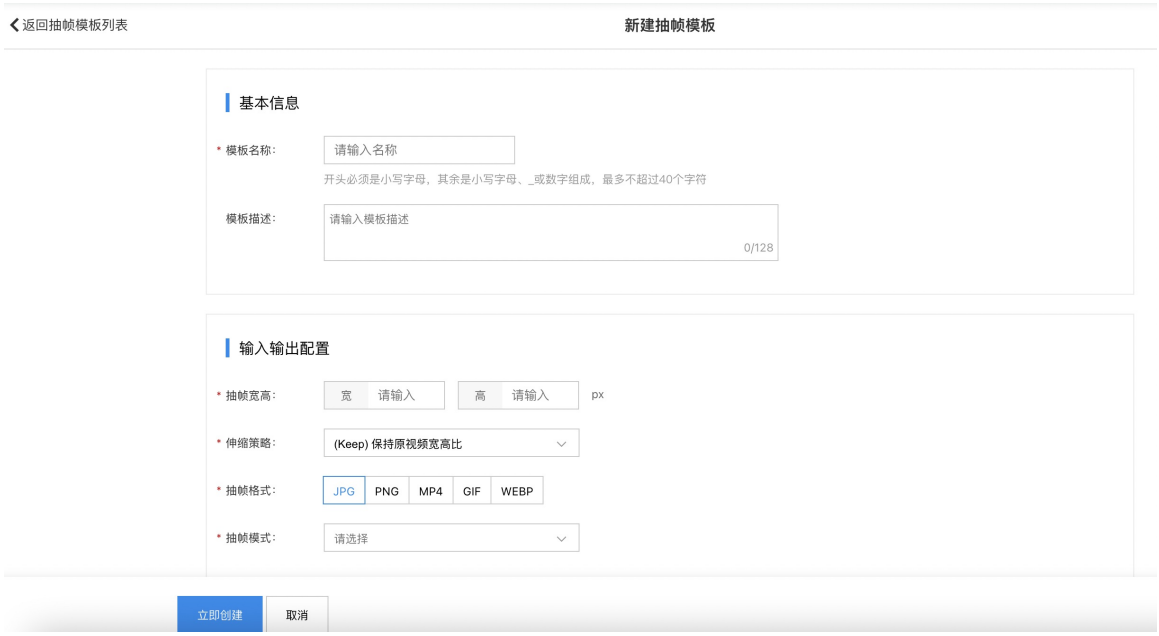
创建抽帧模板

概述

当一系列抽帧任务所设定的参数一致时，可设置抽帧模板。在创建抽帧任务时选择对应的抽帧模板即可。

操作步骤

1. 登录音视频处理 MCP 控制台。
2. 在左侧导航栏选择 **视频抽帧->抽帧模板**，进入“抽帧模板列表”页面。
3. 点击**新建抽帧模板**，进入“新建抽帧模板”页面。



4. 您可以根据需求填写相应内容，点击**立即创建**，完成创建抽帧模板操作。

配置	输入
缩略图宽高	输入宽高值，单位px。
伸缩策略	(Keep) 保持原视频宽高比、(shrinkToFit) 保持原始视频宽高比并加黑边、(stretch) 拉伸原始视频。
缩略图格式	JPG、PNG、MP4、GIF、WEBP 五种格式。
抽帧模式	Auto：系统自动截取熵值较高的一帧； IDL：百度智能算法截取缩略图； Shot：根据场景切换自动截取缩略图； Split：多张，指定起止时间和张数截取； Manual：多张，指定起止时间和间隔时间截取； Highlight：自动生成一个精彩片段六种模式。

> 说明：
>
> 视频截图支持指定截图、智能截图、雪碧图、图片格式、图片宽高、伸缩策略、去水印、黑边剪裁等功能，详见文档 MCP 功能特性。

查看抽帧模板

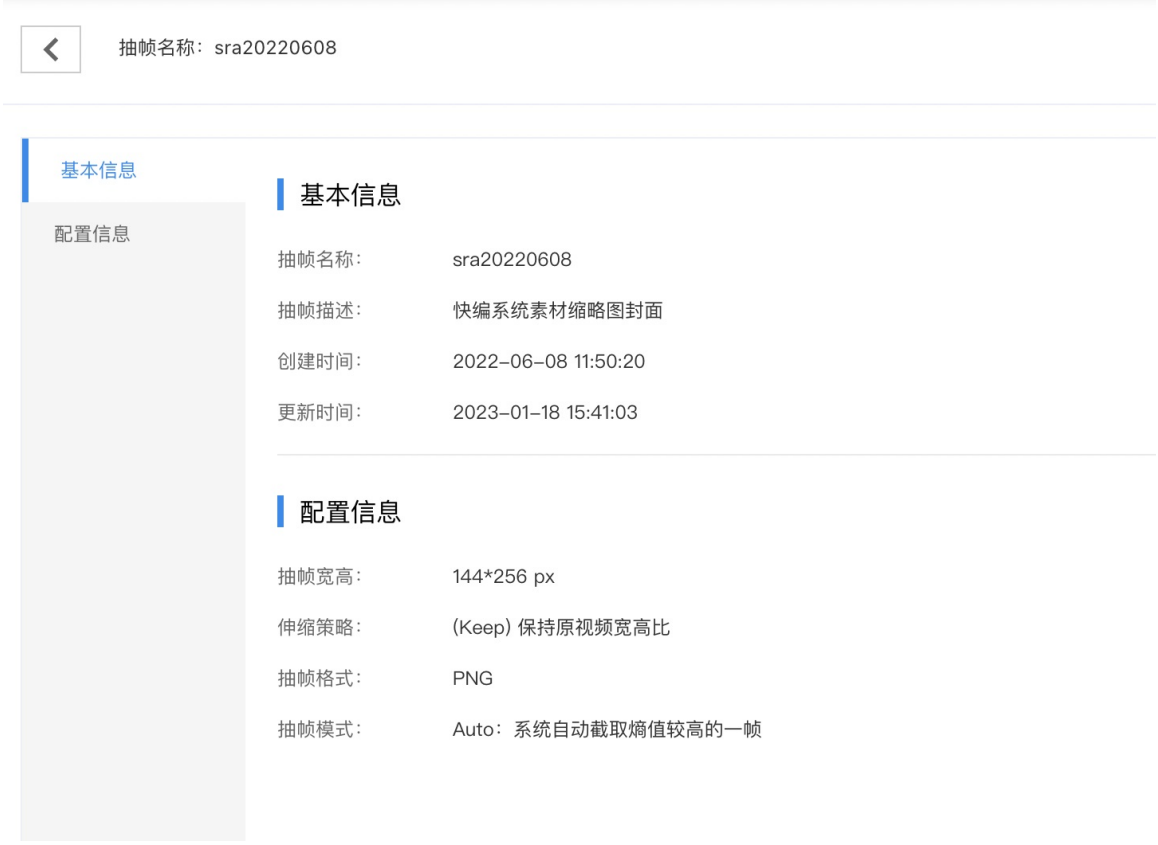
概述

创建抽帧模板后，用户可以在抽帧模板列表页面查看抽帧模板的详细信息。

查看缩略图模板

1. 登录音视频处理 MCP 控制台。
2. 在左侧导航栏选择 **视频抽帧->抽帧模板** ,进入抽帧模板列表页面。
3. 找到目标“模板名称”，点击其操作列的**查看详情**按钮。

4. 在抽帧模板详细信息页面，可以查看抽帧任务的**基本信息**和**配置信息**。



相关 API

- 查询指定抽帧任务

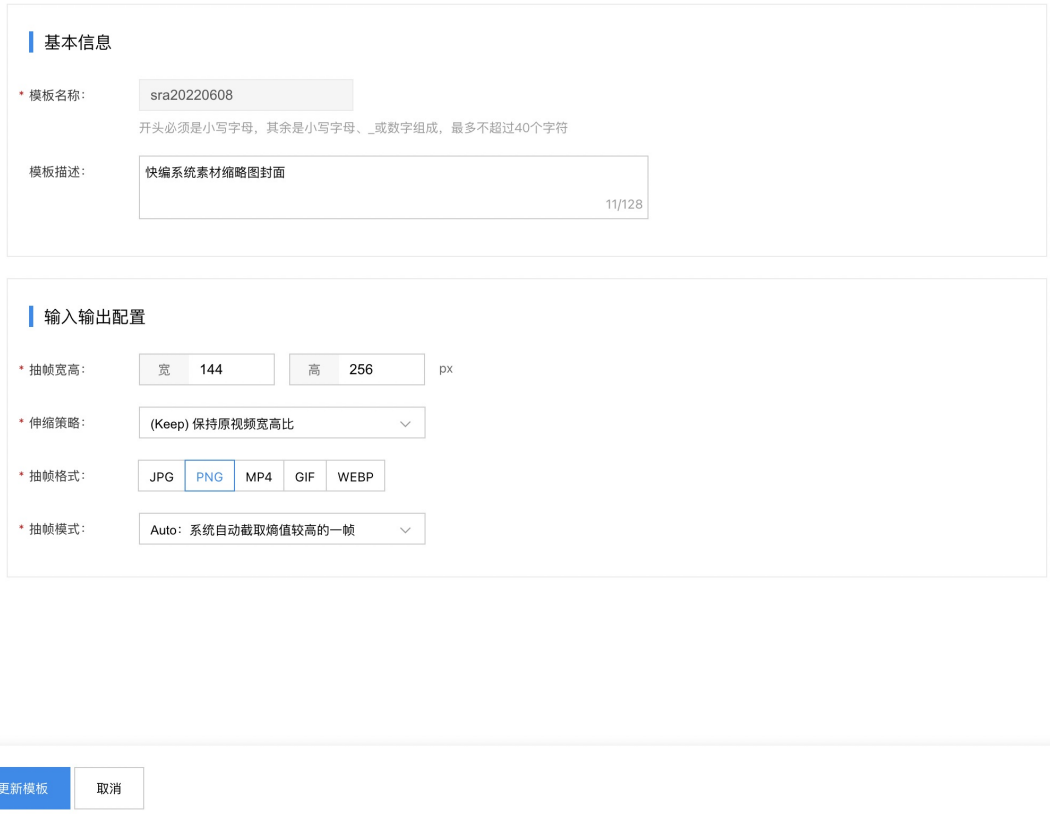
编辑抽帧模板

概述

当模板参数有变更时，可通过编辑抽帧模板来修改模板参数。

操作步骤

1. 登录音视频处理 MCP 控制台。
2. 在左侧导航栏选择 **视频抽帧->抽帧模板** ,进入“抽帧模板”页面。
3. 点击操作列的**编辑**按钮，进入“更新抽帧模板”页面。
4. 除了模板名称不可修改，其他信息均可变更。



5. 变更完成后，点击**更新模板**，即可完成变更操作。

删除抽帧模板

概述

当模板不再使用时，可以通过删除功能删除抽帧模板，一旦删除不可恢复。

操作步骤

1. 登录音视频处理 MCP 控制台。
2. 在左侧导航栏选择 **视频抽帧->抽帧模板** ,进入“抽帧模板列表”页面。
3. 点击操作列的**删除**按钮，弹出删除窗口。

4. 点击**确认**即可删除，一旦删除不可恢复。

质检模板

创建质检模板

概述

如果系统预设的Preset无法满足用户的需求，用户可以自定义自己的Preset。根据不同的转码需求，可以使用不同的接口创建

Preset。

创建自定义质检模板

1. 登录音视频处理 MCP 控制台。
2. 在左侧导航栏选择 **视频质检->质检模板** ,进入质检模板页面。
3. 点击**新建质检模板**，进入“新建质检模板”页面。

4. 用户根据需求填写相应配置信息：

参数	说明
模板名称	开头必须是小写字母，其余可以是小写字母或数字组成，最多不超过40个字符。
模板描述	（可选）输入模板描述。
质检参数	• 质量问题：选择目标检测问题，常见画面问题有：模糊、噪声、马赛克等，常见的音频问题有：静音、音量过低/过高、声音间断等。开启检测画面问题后自定义调整阈值、持续时长、检测间隔； • 阈值：每个检测条件均可支持自定义设定相应强度阈值，可调节范围为0～1，阈值越高，对异常程度评价要求越高； • 持续时长：指定每个检测问题持续的时长，当实际输出值大于/等于设定的持续时长，则被机器自动判定为出现该质量问题； • 检测间隔：指定间隔多长时间检测一帧画面，最长不超过原视频时长。

5. 点击**立即创建**，完成自定义质检模板的创建。

编辑质检模板

概述

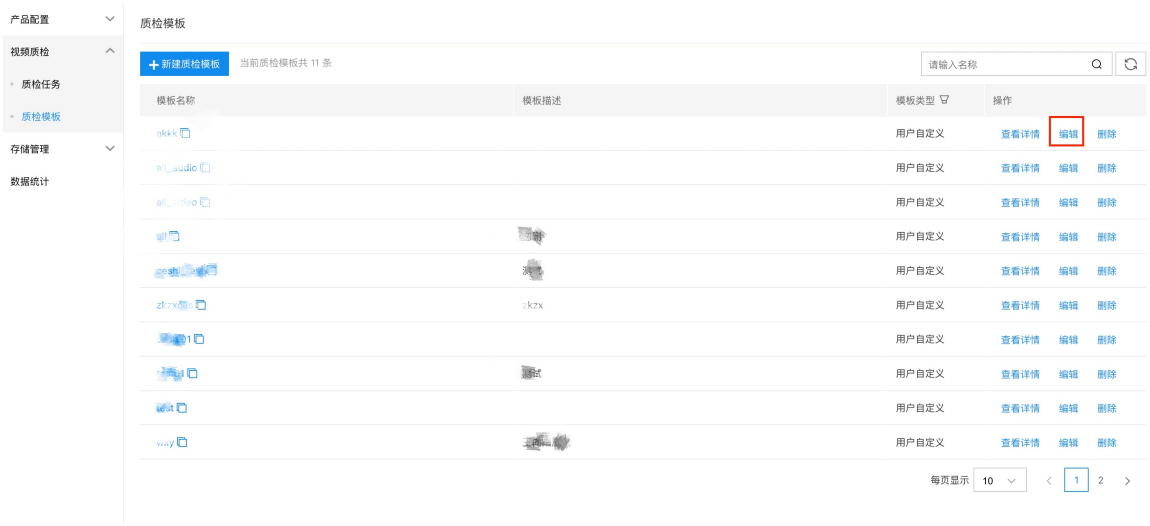
您可以根据不同的检测需求编辑质检模板，修改已创建的自定义质检模板。

编辑质检模板

1. 登录音视频处理 MCP 控制台。

2. 在左侧导航栏选择 **视频质检->质检模板** ，进入质检模板页面。

3. 找到目标“模板名称”，点击其操作列的**编辑**按钮。



4. 在更新质检模板页面，您可以修改模板描述、检测条件配置等配置信息：

5. 修改完成后点击**更新模板**，页面提示“质检模板更新成功”即更新模板成功。

查看质检模板

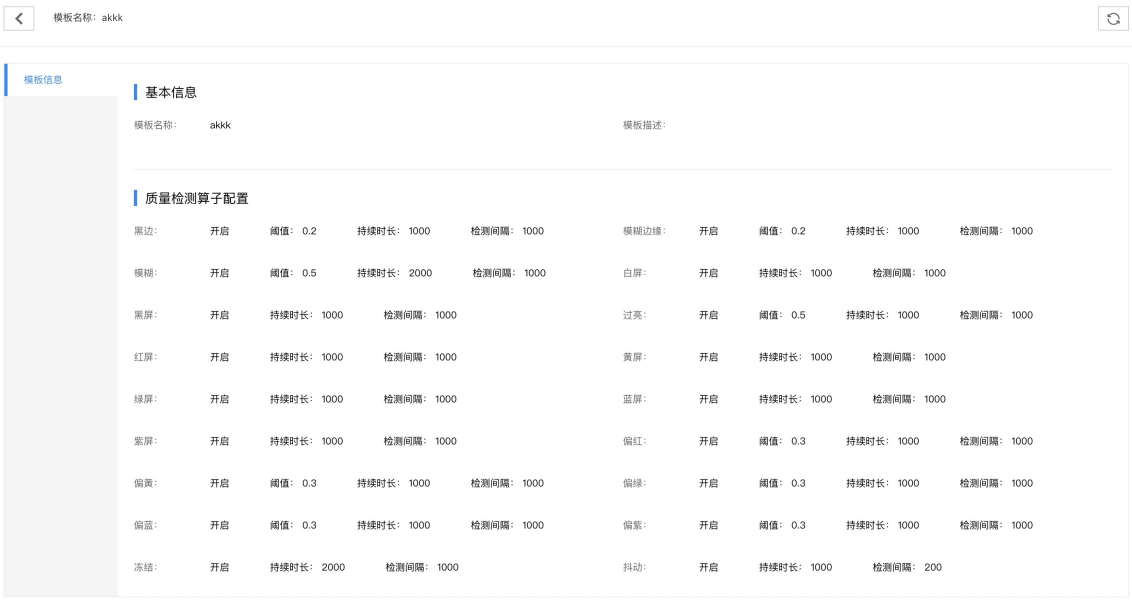
查看质检模板

1. 登录音视频处理 MCP 控制台。

2. 在左侧导航栏选择 **视频质检->质检模板** ,进入质检模板页面。

3. 找到目标“模板名称”，点击其操作列的**查看详情**按钮。

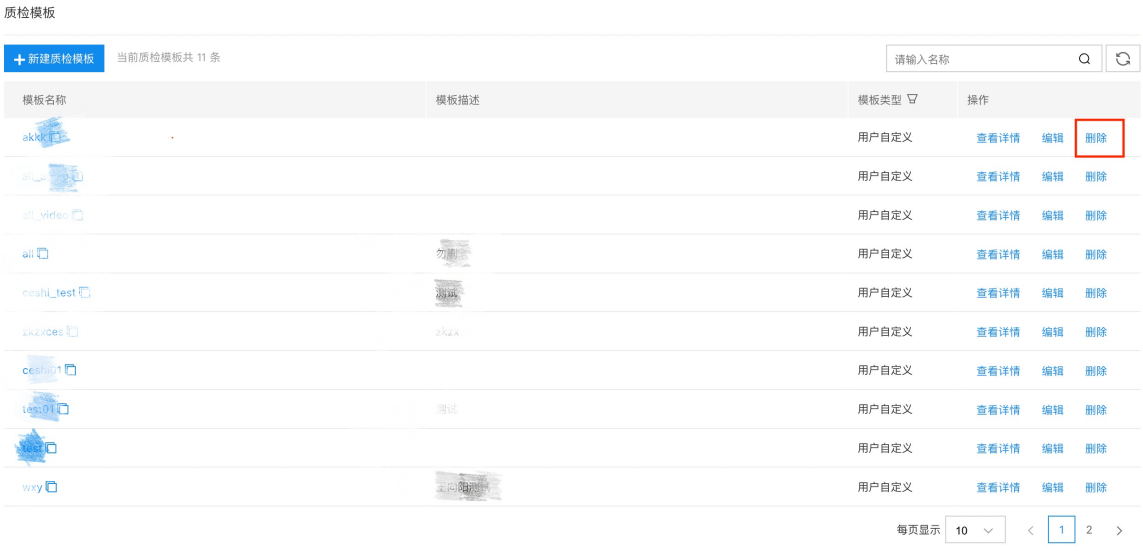
4. 进入模板信息页面，在此页面可以查看**基本信息**、**质量检测算子（检测条件）配置**等信息。



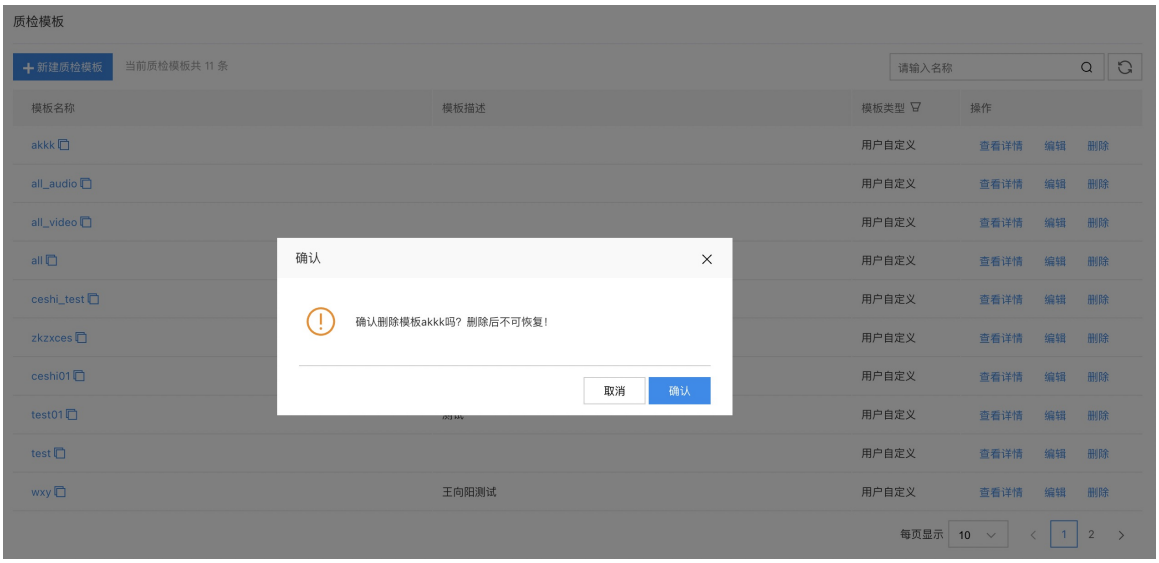
删除质检模板

删除质检模板

1. 登录音视频处理 MCP 控制台。
2. 在左侧导航栏选择 **视频质检->质检模板** ，进入质检模板页面。
3. 通过“模板名称”找到目标模板，点击其操作列的**删除**按钮。



4. 在弹出的确认框中点击**确定**，**删除后将不能恢复，请谨慎操作**



通知管理

概述

本文将向您介绍通知管理的相关操作，包括新建通知、编辑通知、删除通知。

新建通知

创建队列时，用户可以选择配置消息通知，以便随时了解任务状态。用户可以选择不通知，也可以选择消息通知。如果给队列设置了消息通知，那么该队列下的任务结束时会自动将任务状态回调给通知里设置的地址。

消息通知采用HTTP POST推送的形式。

HTTP POST消息格式

```
 POST /notexist HTTP/1.1
host: multimedia.bce-testinternal.baidu.com
accept: _/_
content-type: application/json
content-length: 558

{
 "messageBody": "{\"jobId\":\"job-
nitg7mbbcgisy26a\",\"pipelineName\":\"yourpipeline\",\"jobStatus\":\"SUCCESS\",\"createTime\":\"2022-09-
28T03:27:09Z\",\"startTime\":\"2022-09-28T03:27:09Z\",\"endTime\":\"2022-09-28T03:27:19Z\",\"source\":
{\"clips\":[{\"bucket\":\"yourbucket\",\"sourceKey\":\"test.mp4\"}]},\"target\":
{\"targetBucket\":\"yourtargetbucekt\",\"targetKey\":\"clbktest.mp4\",\"presetName\":\"presetname\",\"adaptiveStrean
:{\"video\":
{\"durationInSeconds\":15,\"sizeInKiloByte\":585.416992,\"widthInPixel\":544,\"heightInPixel\":308,\"frameRate\":30,\
:{\"sampleRateInHz\":22050,\"channels\":2},\"bitRateInKBps\":\"312.22\"}}",
 "messageId": "720b0024-6faf-47ba-91e8-a575665164b7",
 "notification": "notificationname",
 "server": "media.bj.baidubce.com",
 "subscriptionName": "",
 "version": "",
 "signature": ""
}
```

> 说明：用户只需要提取messageBody的内容进行解析。

如果通知采用了SIGN鉴权模式，会在header中增加Notification-Auth-Expire、Notification-Auth-User和Notification-Auth-Token用

于验证。

用户接收到回调后可以使用SHA256-HEX传入自己的token以及"POST;endpoint;content;expireTime;user"生成Notification-Auth-Token，与header中的对比进行验证。

1. token和endpoint分别是您在MCP控制台通知管理添加的签名验证token和回调地址；
2. content是MCP服务端回调您的回调地址时携带的responseBody内容，您在回调地址服务中直接获取即可，无需额外转换(如json转换可能会打乱key的顺序导致签名失败)；
3. expireTime和user是对调请求的请求头中的Notification-Auth-Expire、Notification-Auth-User值。

您可以参考如下Java代码示例

```
public void callbackAndVerify(@RequestBody String object, HttpServletRequest request) {
        System.out.println("callbcak data: " + object);

        String authExpire = request.getHeader("Notification-Auth-Expire");
        System.out.println("Notification-Auth-Expire: " + authExpire);
        String authUser = request.getHeader("Notification-Auth-User");
        System.out.println("Notification-Auth-User: " + authUser);
        String authToken = request.getHeader("Notification-Auth-Token");
        System.out.println("Notification-Auth-Token: " + authToken);

        String token = "yourSecretToken";
        String endPoint = "yourEndpoint";
        String encrypString = EncryAndDecryUtils.getHmacSha256Encrypt(token, "POST;" + endPoint + ";" + object + ";" +
authExpire + ";" + authUser);
        System.out.println("encrypString: " + encrypString);
        if (authToken.equals(encrypString)) {
            System.out.println("verify pass");
        }
    }
```

sha256加密方法：

```
public static String getHmacSha256Encrypt(String secret, String message) {
        String encryptString = "";
        try {
            Mac mac = Mac.getInstance("HmacSHA256");
            SecretKeySpec secretKey = new SecretKeySpec(secret.getBytes(CHARSET_UTF8), "HmacSHA256");
            mac.init(secretKey);
            byte[] hash = mac.doFinal(message.getBytes(CHARSET_UTF8));
            return new String(Hex.encodeHex(hash));
        } catch (Exception e) {
            e.printStackTrace();
        }
        return encryptString;
    }
```

操作步骤

1. 登录音视频处理 MCP 控制台。
2. 在左侧导航栏选择 **产品配置->通知管理** ，进入“通知管理”页面。
3. 点击**新建通知模板**，进入“新建通知”页面。
4. 填写通知名称、回调地址和回调验证类型，使用签名验证时，需要输入一个Token，使用方法参考通知接口API。

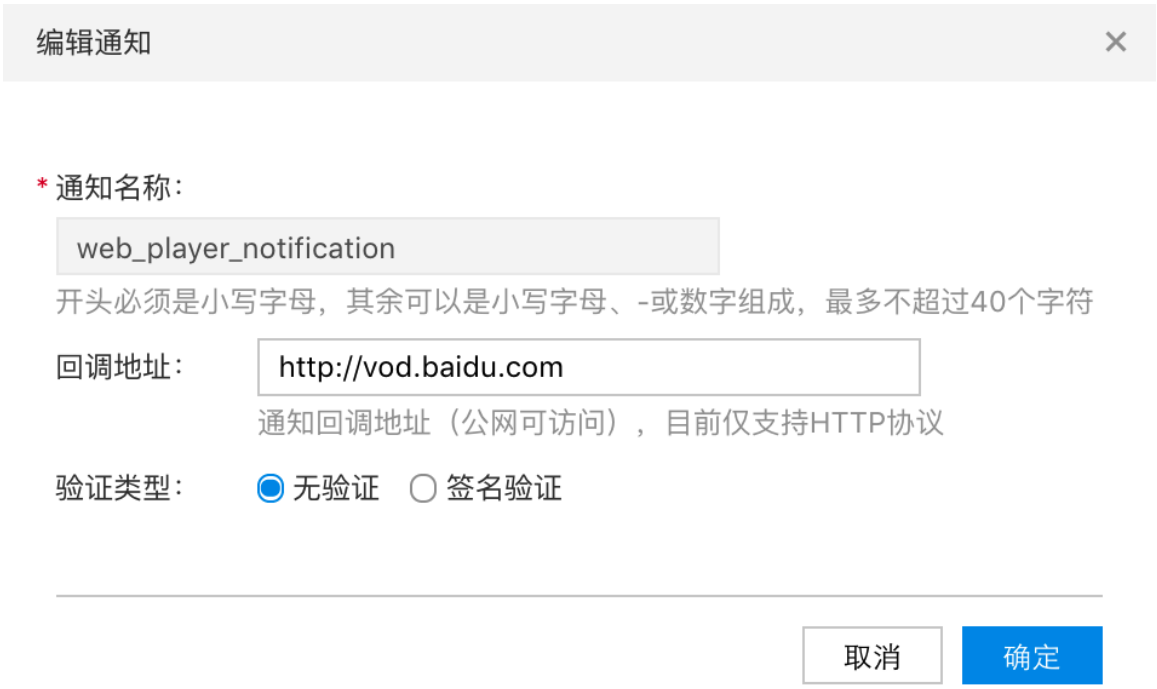
5. 填写好相关信息，点击**确定**，完成创建通知模板操作。

编辑通知

通知配置信息有变时，可以通过编辑通知配置更新通知信息。

操作步骤

1. 登录音视频处理 MCP 控制台。
2. 在左侧导航栏选择 **产品配置->通知管理** ,进入“通知管理”页面。
3. 点击操作列中的**编辑**按钮，进入“编辑通知”页面。
4. 除了通知名称不可编辑，其他信息均可编辑。



5. 确认好相关信息，点击**确定**，完成修改通知模板操作。

删除通知

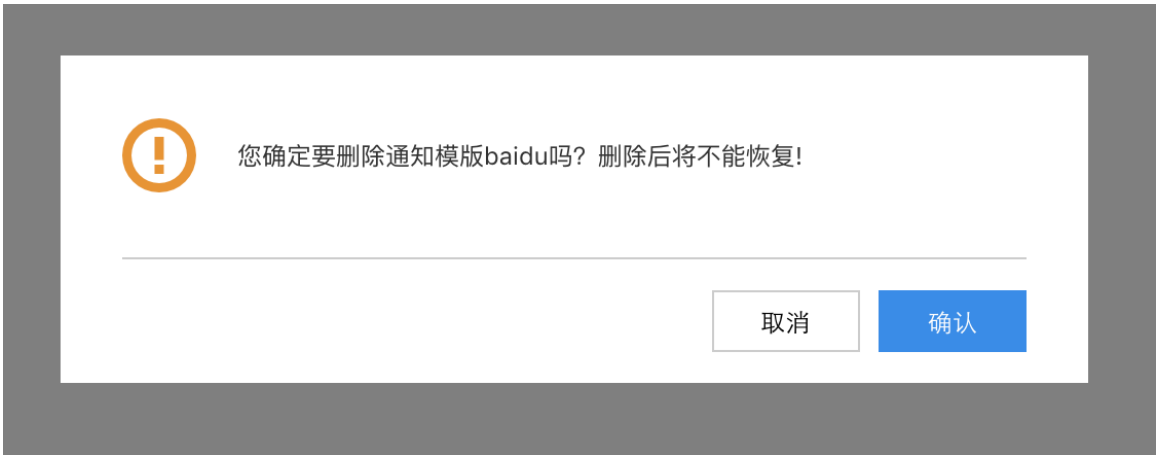
通知配置不使用的情况下，可以删除通知。

注意事项

通知配置一旦删除不能恢复。

操作步骤

1. 登录音视频处理 MCP 控制台。
2. 在左侧导航栏选择 **产品配置->通知管理** ,进入“通知管理”页面。
3. 点击操作列中的**删除**按钮，弹出删除窗口。



4. 点击**确定**，完成删除通知模板操作，一旦删除不能恢复。

相关API

- 通知接口API

创建转码任务

概述

本文将向您介绍转码任务的相关操作，包括创建转码任务、查看任务详情。

前提条件

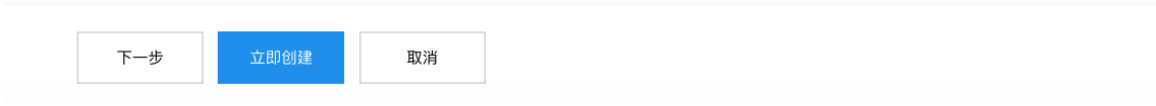
- 创建转码任务前，请先确认已经创建任务队列，并且将要处理的视频上传与存储到BOS中。
- 创建转码任务前必须已添加模板，转码模板会有系统默认模板，用户也可以自定义模板。

创建转码任务

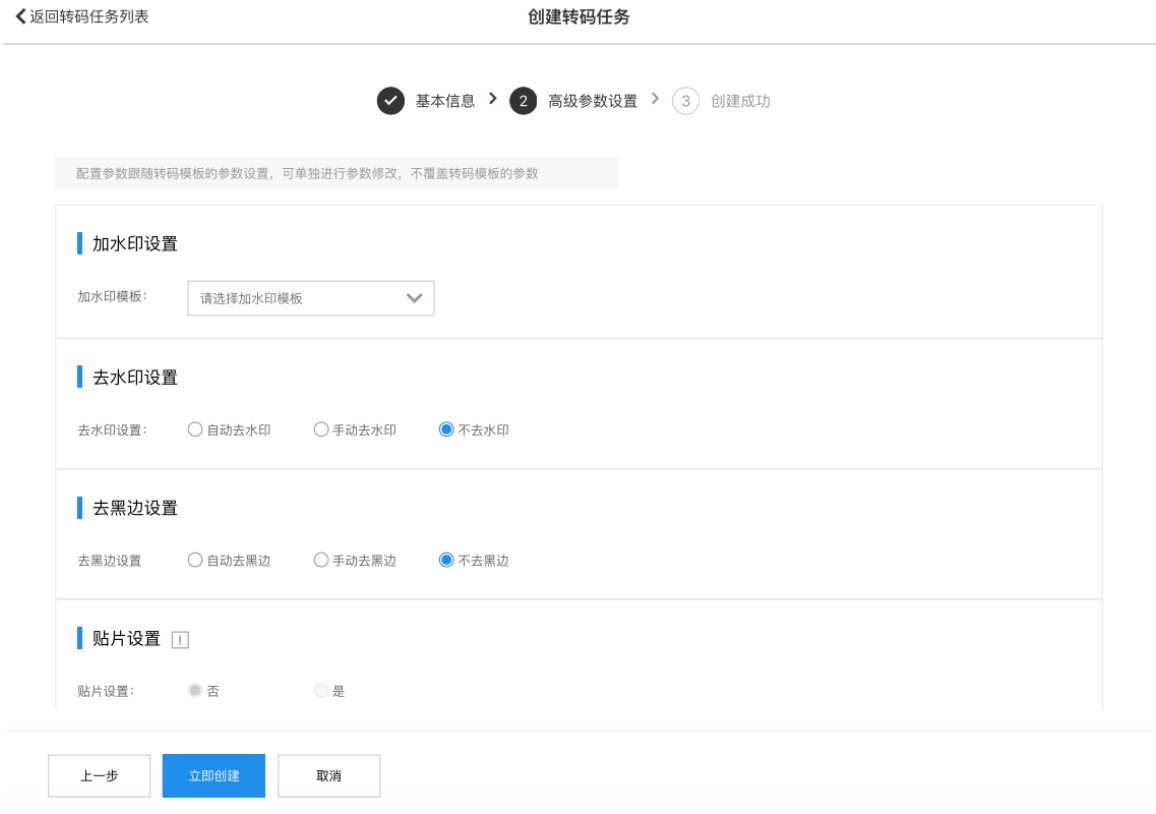
确认好已经创建任务队列，并且已将要处理的视频上传与存储到BOS以后，进行创建转码任务的操作。

操作步骤

1. 登录音视频处理 MCP 控制台。
2. 在左侧导航栏选择 **任务管理->转码任务** ,进入“转码任务”页面。
3. 点击**新建转码任务**，进入“新建转码任务”页面。
4. 在“新建转码任务”页面，进行基本参数的配置，选择相应的任务队列、转码输入文件、转码模板，填写输出文件名。



5. 如不需高级参数配置，可点击**立即创建**按钮，任务创建成功，进入任务列表。
6. 如需配置高级参数，点击**下一步**按钮，进入高级参数的配置。
   - 加水印设置：可以选择添加水印，支持多个水印添加。
   - 去水印设置：可以选择自动去水印，或者手动输入水印坐标位置进行去水印。
   - 去黑边设置：可以选择自动去黑边，或者手动输入有效视频的坐标位置进行去黑边。
   - 贴片设置：当使用模板的尺寸伸缩策略为：自适应伸缩加黑边(shrinkToFit)时，可开启贴片设置功能。给视频设置前后贴片信息。



7. 完成高级参数配置，点击**立即创建**按钮，任务创建成功，进入任务列表。

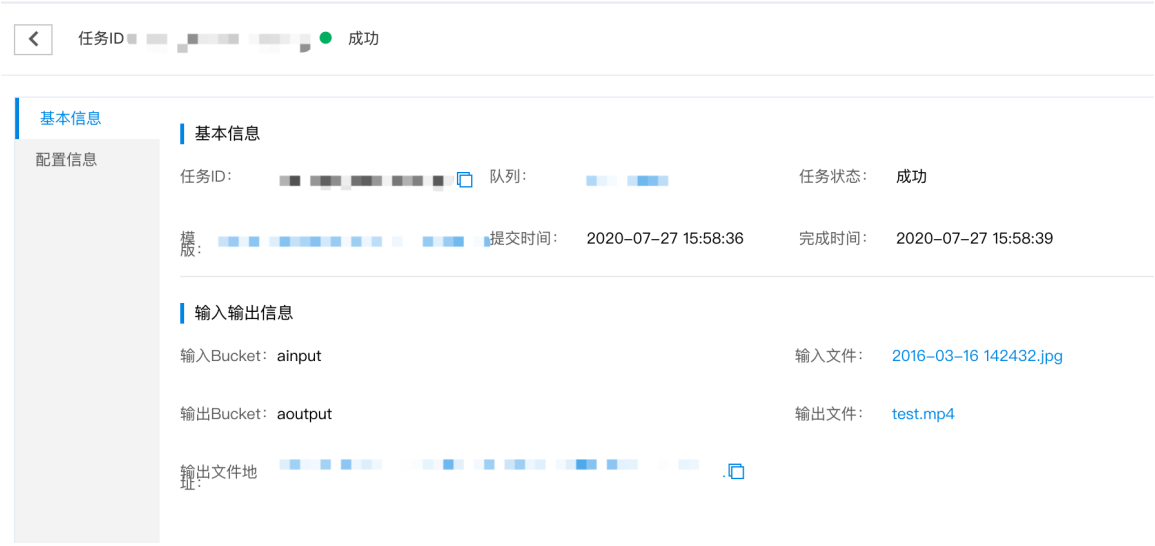
查看任务详情

任务创建成功后，用户可以在任务详情页看到任务的详细信息。包括任务的运行状态、基础信息、输入输出信息、配置等信息。

操作步骤

1. 登录音视频处理 MCP 控制台。
2. 在左侧导航栏选择 **任务管理->转码任务** ,进入“转码任务”页面。
3. 点击“任务ID”列的**ID链接**或者“操作”列的**查看详情**按钮，进入“转码任务详情”页面。

4. “转码任务详情”页面，可查看任务“基本信息”和“配置信息”。



相关 API

- 视频转码任务接口
- 水印接口

创建抽帧任务

概述

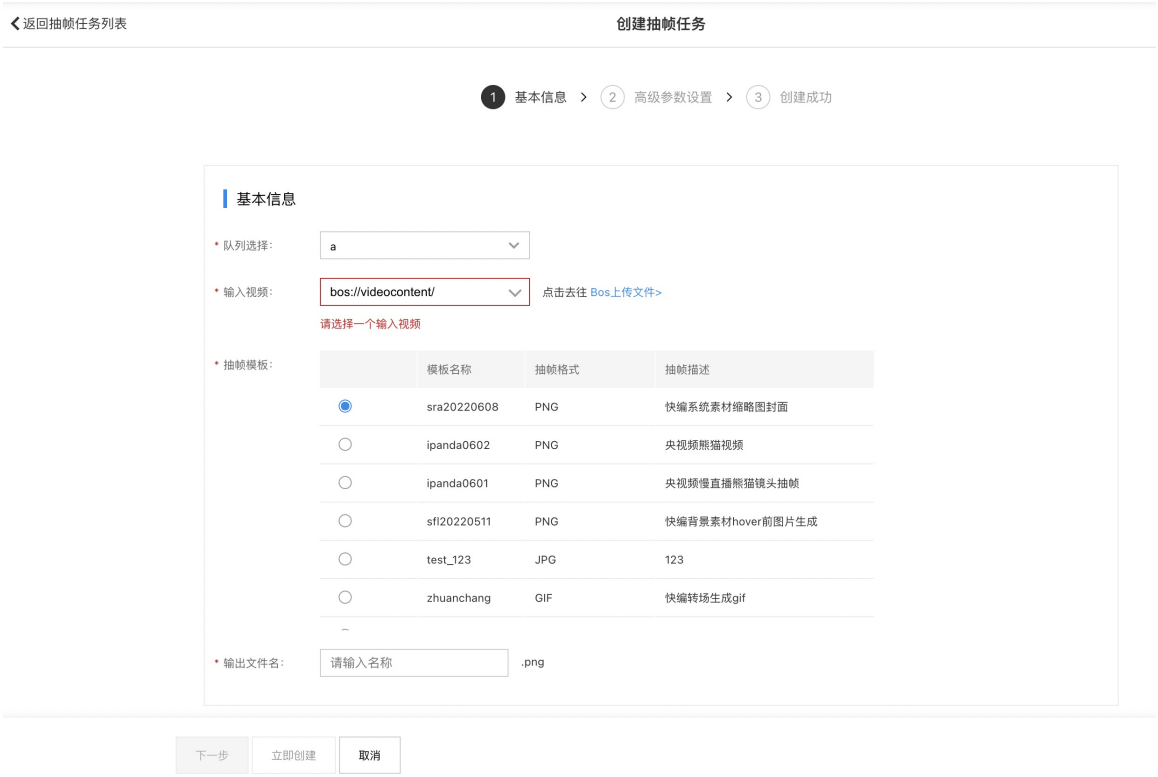
本文将向您介绍抽帧任务的相关操作，包括创建抽帧任务、查看任务详情。

前提条件

- 创建抽帧任务前必须添加抽帧模板，抽帧模板无系统默认的模板，用户必须自行添加。
- 创建抽帧任务前，请先确认已经创建任务队列，并且将要处理的视频上传与存储到BOS中。

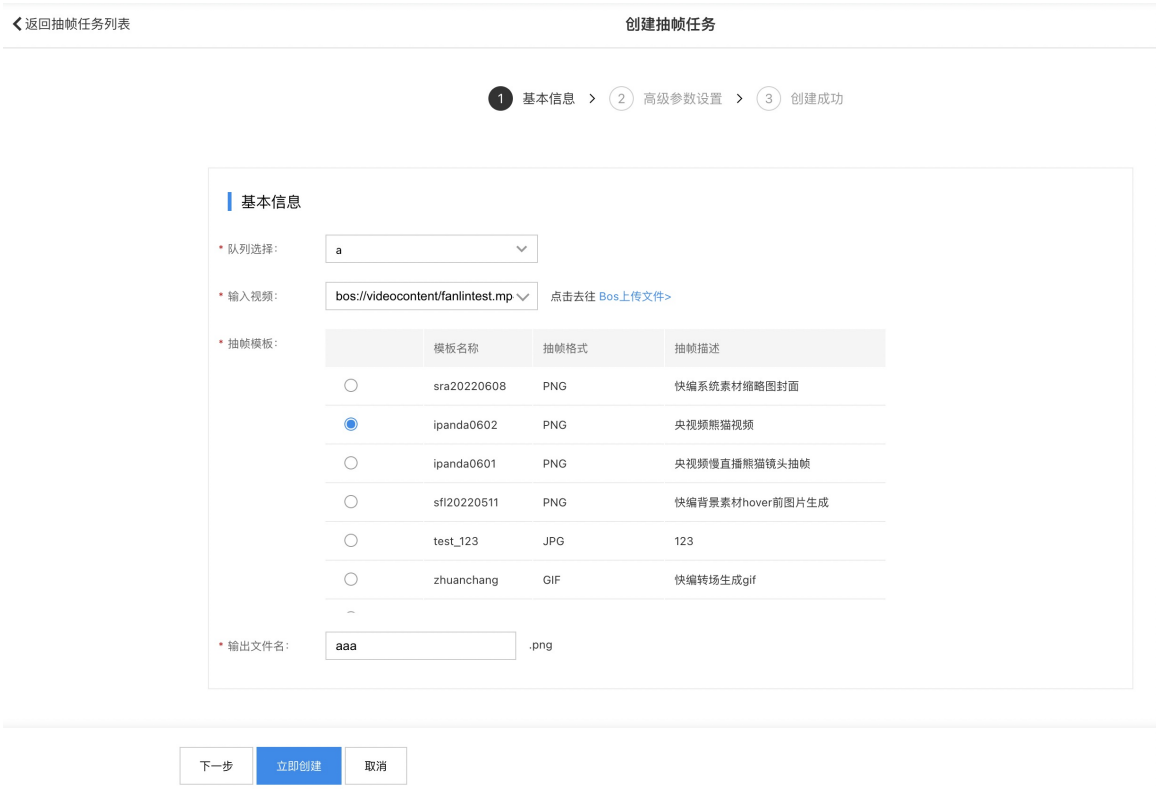
创建抽帧任务

确认好已经创建任务队列，并且已将要处理的视频上传与存储到BOS以后，进行创建抽帧任务的操作。



操作步骤

1. 登录音视频处理 MCP 控制台。
2. 在左侧导航栏选择 **视频抽帧->抽帧任务** ,进入“抽帧任务”页面。
3. 点击**新建抽帧任务**，进入“创建抽帧任务”页面。
4. 在“创建抽帧任务”页面，进行基本参数的配置，选择相应的任务队列、输入视频、抽帧模板，填写输出文件名。



5. 如不需高级参数配置，可点击**立即创建**按钮，任务创建成功，进入任务列表。
6. 如需配置高级参数，点击**下一步**按钮，进入高级参数的配置。

- 去水印设置，可以选择自动去水印，或者手动输入水印坐标位置进行去水印。
- 去黑边设置，输入有效视频的坐标位置进行去黑边。

7. 完成高级参数配置，点击**立即创建**按钮，任务创建成功，进入任务列表。

查看任务详情

任务创建成功后，用户可以在任务详情页看到任务的详细信息。包括任务的运行状态、基础信息、输入输出信息、配置等信息。

操作步骤

1. 登录音视频处理 MCP 控制台。
2. 在左侧导航栏选择 **视频抽帧->抽帧任务** ,进入“抽帧任务”页面。
3. 点击“任务ID”列的**ID链接**或者“操作”列的**查看详情**按钮，进入“抽帧任务详情”页面。

4. "抽帧任务详情”页面，可查看任务“基本信息”和“配置信息”。

相关 API

- 抽帧任务接口

CDN加速

概述

用户转码输出的视频文件存储在BOS上，可以和普通的文件操作一样设置CDN加速操作。CDN和BOS针对视频文件做过专门的优化，支持range参数，可以更好的保证视频文件的分发效率，同时降低流量开销。

开通 CDN 官方加速域名

1. 登录 BOS 管理控制台，进入全局概览页面。
2. 在 BOS 控制台左侧的导航栏中选择**目标 Bucket**，点击打开该 Bucket 的文件列表页。
3. 点击页面上方导航栏的**发布管理**页签，进入发布管理页面。
4. 点击“官方域名发布“模块下的 **CDN 加速开关**，即可直接开启官方的 CDN 加速域名。
5. 开启加速成功后，刷新页面可以看到 CDN 加速下的加速域名后出现**管理 CDN 域名**按钮，点击可以进入 CDN 的域名管理页面对域名详情进行管理。

相关文档

- CDN加速发布

播放器

转码任务输出的标准视频，可以使用百度提供的播放器，也可以使用任意的支持MP4/FLV/流媒体格式的播放器播放。对于加密的视频（通过API方式可以实现转码输出AES-128加密的流媒体视频），需要采用支持AES-128解密的视频播放器播放。

百度提供如下播放器 SDK：

- Web平台
  - 播放器Web SDK帮助手册
  - 播放器Web SDK下载
- Android平台
  - 播放器Android SDK帮助手册
  - 播放器Android SDK下载
- iOS平台
  - 播放器iOS SDK帮助手册
  - 播放器iOS SDK下载

创建质检任务

概述

本文将向您介绍质检任务的相关操作，包括创建质检任务、查看任务详情。

前提条件

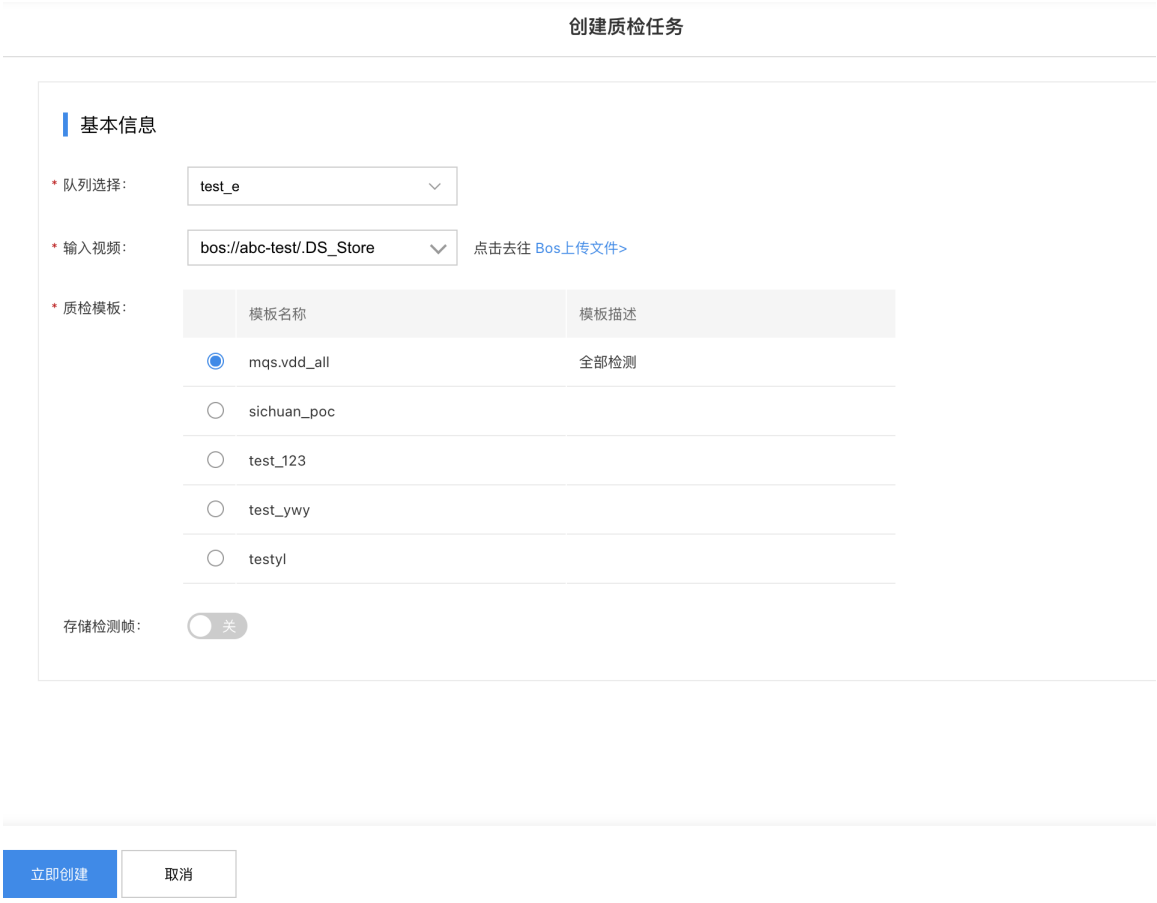
- 创建质检任务前，请先确认已经创建任务队列，并且将要处理的视频上传与存储到BOS中。
- 创建质检任务前必须已添加模板，质检模板会有系统默认模板，用户也可以自定义模板。

创建转码任务

确认好已经创建任务队列，并且已将要处理的视频上传与存储到BOS以后，进行创建转码任务的操作。

操作步骤

1. 登录音视频处理 MCP 控制台。
2. 在左侧导航栏选择 **视频质检->质检任务** ,进入“质检任务列表”页面。
3. 点击**新建质检任务**，进入“创建质检任务”页面。
4. 在“创建质检任务”页面，选择相应的任务队列、质检输入文件、质检模板，选择是否存储检测帧图片（检测图片自动存储BOS-bucket固定路径下）。



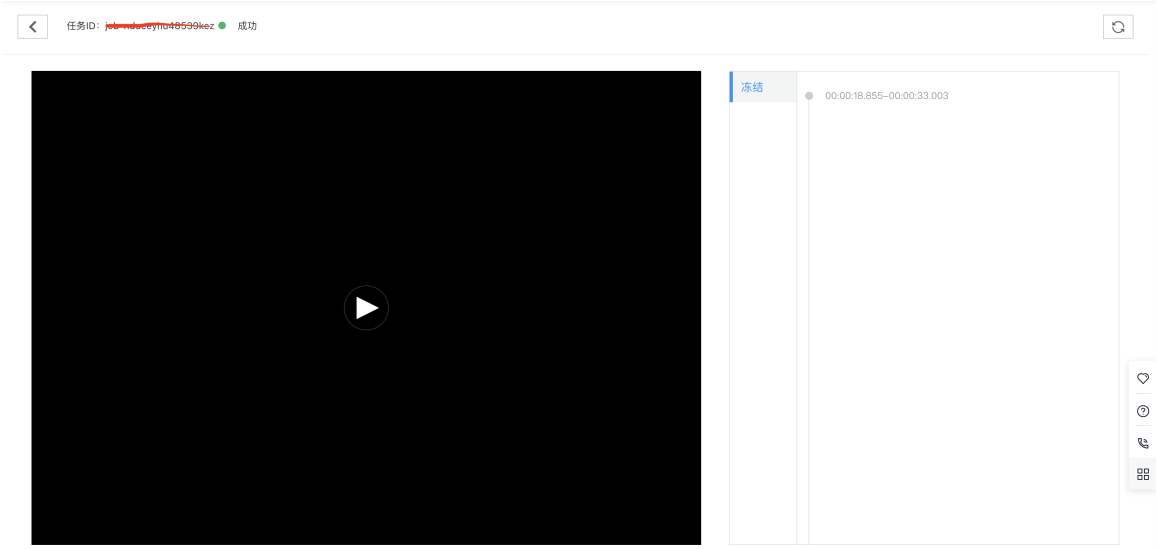
5. 点击**立即创建**按钮，任务创建成功，进入质检任务列表。

查看任务详情

任务创建成功后，用户可以在任务详情页看到任务的详细信息。包括任务ID、任务基础信息、任务的运行状态、可配置操作：查看详情、重新处理、结果导出。

操作步骤

1. 登录音视频处理 MCP 控制台。
2. 在左侧导航栏选择 **视频质检->质检任务** ,进入“质检任务列表”页面。
3. 点击“任务ID”列的**ID链接**或者“操作”列的**查看详情**按钮，进入“转码任务详情”页面。
4. “转码任务详情”页面，可查看任务ID、输入原视频预览播放、出现的质量问题名称和时间段（精确到三位小数点的秒数）。



相关 API

- 视频质量检测任务接口
- 视频质量检测模板接口

数字水印

图片嵌入水印

概述

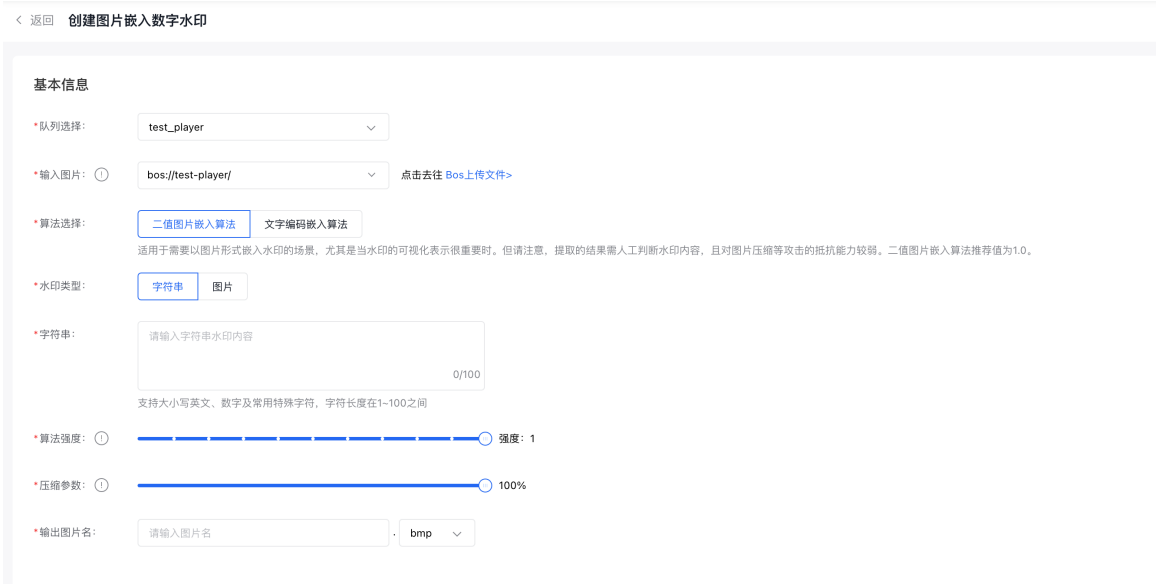
音视频处理可以自定义图片嵌入数字水印，您可通过本文档了解如何使用控制台进行图片嵌入数字水印。

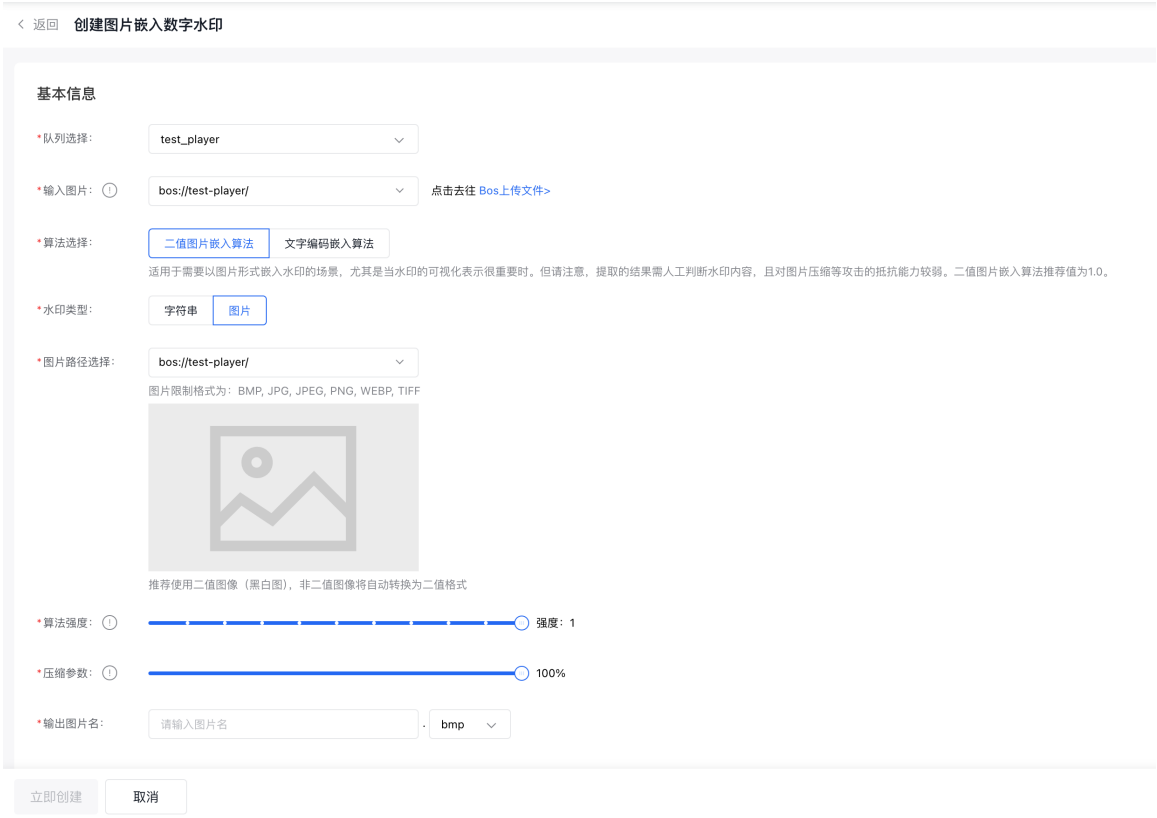
前提条件

- 数字水印字符串格式支持1~100个字符，支持英文、数字及常用特殊字符。
- 数字水印图片限制格式为：PNG, JPG, BMP, JPEG格式。
- 关于如何将水印图片上传至BOS，请参考上传Object。

创建图片嵌入水印任务

1. 登录音视频处理 MCP 控制台。
2. 在左侧导航栏选择 **数字水印->图片嵌入水印**，进入图片嵌入数字水印任务页面。
3. 点击 **图片嵌入水印** 按钮，进入新建任务页面。





4. 用户根据需求填写相应配置信息：

配置项	**说明**
队列选择	从下拉列表中选择适用的处理队列。
输入图片	选择存储在BOS内的图片地址作为待处理图片。
算法选择	• 二值图片嵌入算法：适用于需要以图片形式嵌入水印的场景。请注意，提取的结果需人工判断水印内容，且对图片压缩等攻击的抵抗能力较弱； • 文字编码嵌入算法：适用于需要以文字形式嵌入水印的场景。该算法支持加密，确保只有授权用户能提取水印，适用于对安全性有高要求的应用场景。
水印内容	• 字符串水印：输入1~100字符范围内的水印内容，支持英文、数字及常用特殊字符； • 图片水印：通过下拉选择BOS内的图片地址，所选图片将在下方预览。
算法强度	调整算法的强度（0.1~1），值越大，抗攻击性越强，但可能影响图片质量。
压缩参数	设置输出图片的质量（1~100%），压缩参数会影响图片的画质和文件大小： • BMP格式：压缩设置不影响。 • PNG格式：降低压缩参数会减小文件大小，但不影响画质，参数越接近100%影响越小。 • 其他格式：降低压缩参数会减小文件大小并损坏图片画质，参数越接近100%影响越小。
输出图片名	指定输出图片的名称和格式（支持bmp、jpg、jpeg、png、webp、tiff），默认为bmp格式。

5. 点击 **创建任务**，完成图片嵌入数字水印任务的创建。

相关API

- 图片数字水印嵌入接口

图片水印提取

概述

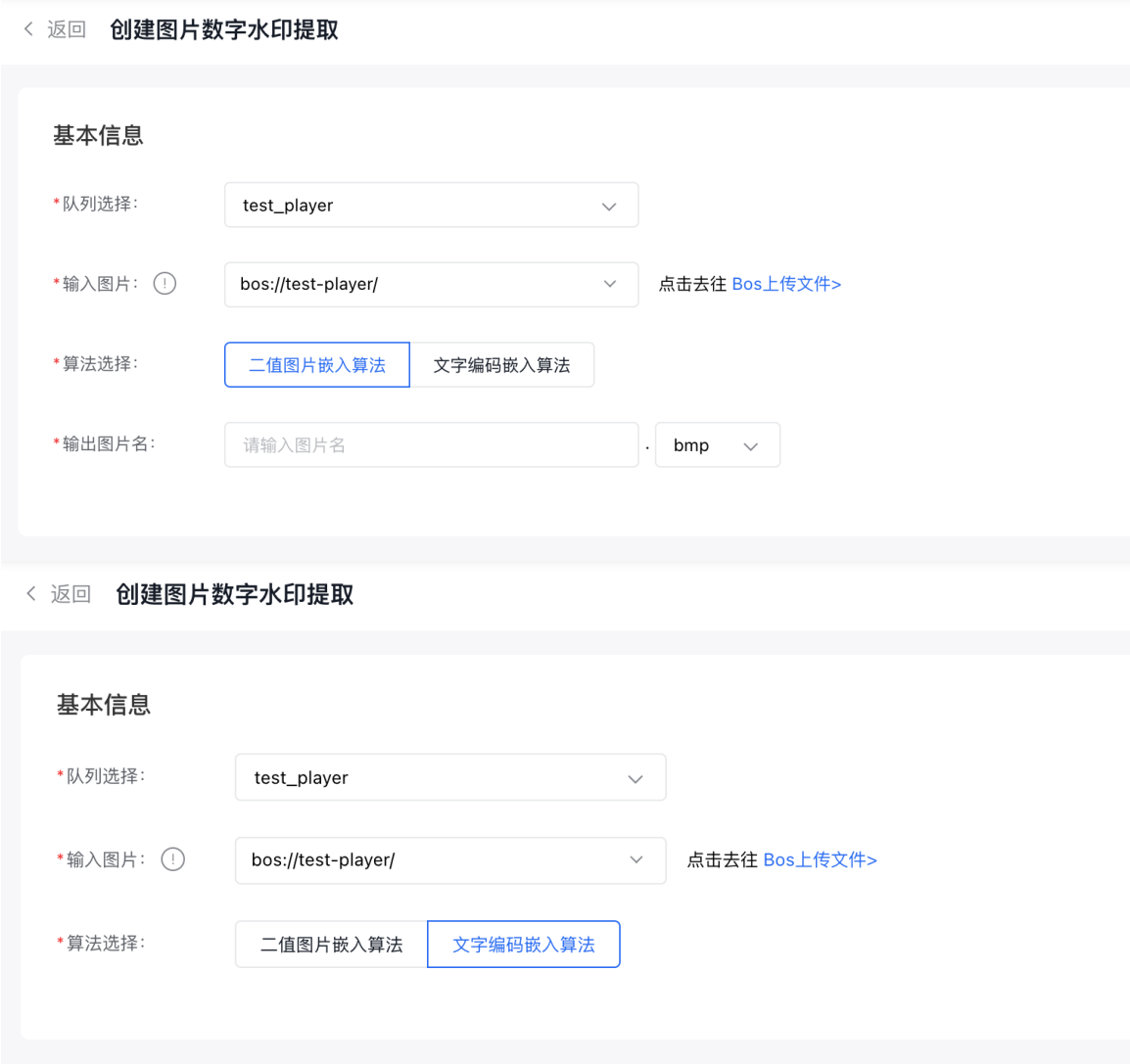
本文档指导您如何通过控制台创建图片水印提取任务，实现对图片文件中嵌入的数字水印的提取。

前提条件

- 关于如何将待提取水印的图片上传至BOS，请参考上传Object。

创建图片水印提取任务

1. 登录音视频处理 MCP 控制台。
2. **访问图片水印提取页面：**在控制台首页点击 **数字水印->图片水印提取**，进入图片水印提取任务页面。
3. 点击 **图片水印提取** 按钮，进入新建任务页面。



4. 用户根据需求填写相应配置信息：

配置项	说明
队列选择	从下拉列表中选择适用的处理队列。
输入图片	选择存储在BOS内的图片地址作为待处理图片。
算法选择	• 二值图片嵌入算法：适用于从图片中提取以图片形式嵌入的水印。提取结果为图片，需要用户填写输出图片名并选择输出格式和地址； • 文字编码嵌入算法：适用于从图片中提取以文本形式嵌入的水印。提取结果为字符串，可在任务详细页面直接查看。
输出设置	仅限二值图片嵌入算法： • 输出图片名：填写提取水印后的输出图片名称，必填项； • 图片格式选择：可选择bmp、jpg、jpeg、png、webp、tiff格式，默认为bmp； • 输出地址：选择输出图片的存储地址，确保易于找到和访问。

5. 点击 **创建任务**，完成图片数字水印提取任务的创建。

注意事项

- 确保所选算法与水印类型匹配，以便正确提取水印。
- 对于二值图片嵌入算法提取的图片水印，务必确认输出图片的格式和存储位置。

相关API

- 图片数字水印提取接口

视频水印模板

概述

音视频处理可以自定义视频数字水印模板，您可通过本文档了解如何使用控制台创建视频数字水印模板。

前提条件

- 数字水印字符串格式支持中英文、特殊字符，字符串长度在4～40之间。
- 数字水印图片限制格式为：PNG, JPG, BMP, JPEG支持png（推荐）、jpg、gif格式。
- 关于如何将水印图片上传至BOS，请参考上传Object。

创建水印模板

1. 登录音视频处理 MCP 控制台。
2. 在左侧导航栏选择 **数字水印->视频水印模板** ,进入视频数字水印模板任务页面。
3. 点击 **数字水印模板**，进入“新建数字水印模板”页面。

4. 用户根据需求填写相应配置信息：

配置项	说明
水印类型	需要选择水印的类型。可选项为“文本”或“图片”: • 若选择“文本”，将提供一个字符串输入框，用于输入想要嵌入的文本内容； • 若选择“图片”，将提供一个选项，用于从BOS地址选择水印图片。选择后，可以预览所选的图片，确保它符合需求。
模版描述	无论选择文本还是图片作为水印，都建议填写模版的描述，这有助于未来更容易地识别和使用该模板。

5. 点击 **确认**，完成数字水印模板的创建。 说明：水印是创建自定义模板的可配置项。

相关API

- 视频数字水印模板接口

视频水印提取

概述

本文档指导您如何通过控制台创建视频水印提取任务，实现对视频文件中嵌入的数字水印的提取。

前提条件

- 关于如何将待提取水印的视频上传至BOS，请参考上传Object。

创建视频水印提取任务

1. 登录音视频处理 MCP 控制台。
2. 在左侧导航栏选择 **数字水印->视频水印提取**，进入视频数字水印提取任务页面。
3. 点击 **视频水印提取**，进入新建视频水印提取页面。
4. 用户根据需求填写相应配置信息：

配置项	说明
队列选择	从下拉列表中选择适用的处理队列。
输入视频	选择存储在BOS内的视频地址作为待处理视频。 （注：不支持HLS、A-HLS、DASH格式视频的嵌入和提取数字水印操作）
检测类型	选择检测类型，包括文本或图片： • 若选择文本，提供数字水印模版ID（可选）和相对应的密钥（可选）； • 若选择图片，需填写输出图片名称和选择输出格式及地址。
数字水印模版ID	若ID已知，输入水印模版ID以指定提取哪一种水印。
密钥	若水印加密，输入相应的密钥以正确提取水印。

5. 点击 创建任务，完成视频数字水印提取任务的创建。

注意事项

- 确保所选检测类型与视频中嵌入的水印类型相匹配。

相关API

- 视频数字水印提取接口

加密密钥管理

概述

本文档指导您如何通过控制台创建和管理数字水印的加密密钥，以增强水印数据的安全性。

创建加密密钥

1. 登录音视频处理 MCP 控制台。
2. 在左侧导航栏选择 **数字水印->加密密钥管理**，进入加密密钥管理页面。
3. 点击 **加密密钥**，进入新建数字水印密钥页面。
4. 用户根据需求填写相应配置信息：

配置项	说明
密钥ID	系统会自动分配一个唯一的密钥ID。
加密密钥填写	输入加密密钥，密钥支持大小写英文、特殊字符，长度需在6～16字符之间。
密钥描述	为创建的密钥提供一个描述，有助于未来识别和管理密钥。

5. 点击 **确认**，完成视频数字水印提取任务的创建。

注意事项

- 密钥描述有助于在管理多个密钥时快速识别特定密钥的用途，建议填写具有辨识度的描述。
- 保管好您的加密密钥，避免泄露给未授权的人员，以保障数据的安全性。

相关API

- 视频数字水印密钥模板接口

API参考

使用须知

使用API访问MCP服务前，请先了解以下内容：

- MCP核心概念
- API认证机制
- 如何获取AKSK
- 区域选择说明

MCP目前支持“华北-北京”、“华南-广州”和“华东-苏州”三个Region，服务域名为：

区域	ID	服务域名	协议
北京	bj	media.bj.baidubce.com	HTTP/HTTPS (为了提升数据的安全性，建议通过HTTPS调用)
广州	gz	media.gz.baidubce.com	HTTP/HTTPS
苏州	su	media.su.baidubce.com	HTTP/HTTPS

如果您是初次调用百度智能云产品的API，可以观看API入门视频指南，快速掌握调用API的能力。也可以通过可视化API调试工具

- 示例代码中心，进行学习与调用测试。

> **注意：**
>
> 1. 由于音视频转码的整体服务的速度，与视频拉取速度有密切关系，因此为了保证整体服务的高效，音视频转码的服务区域会和百度对象存储BOS的服务区域保持一致。例如，北京区域的转码服务只处理北京区域BOS上存储的音视频资源，广州区域的转码服务只能处理广州BOS存储上的音视频资源，以此类推。
> 2. 为了保证 MCP API能够被授权访问 BOS，CDN 等服务，用户需要在管理控制台至少创建一次任务队列。

系统限制

任务并发数的限制

Pipeline类型	Job并发数	Pipeline最大个数
免费型	20	5
私有型	不限	不限

单个原始音视频文件大小限制

单个原始音视频文件的大小限制和BOS单个可上传文件的大小限制保持一致，为5TB。

日期与限制

日期与时间的表示有多种方式。为统一起见，除非是约定俗成或者有相应规范的，凡是HTTP标准中规定的表示日期和时间字段用GMT，其他日期时间表示的地方一律采用UTC时间，遵循ISO 8601，并做以下约束：

- 表示日期一律采用`YYYY-MM-DD`方式，例如`2014-06-01`表示2014年6月1日。
- 表示时间一律采用`hh:mm:ss`方式，并在最后加一个大写字母Z表示UTC时间。例如`23:00:10Z`表示UTC时间23点0分10秒。
- 凡涉及日期和时间合并表示时，在两者中间加大写字母T，例如`2014-06-01T23:00:10Z`表示UTC时间2014年6月1日23点0分10秒。

接口规范

请求头域内容（HTTP Request Header）

音视频转码的API服务需要在请求的HTTP头域中包含以下信息：

- host（必选）
- x-bce-date （必选）
- x-bce-request-id（可选）
- authorization（必选）
- content-type（可选）
- content-length（可选）

作为示例，以下是一个标准的用户获取队列清单时的请求头域内容:

```
GET /v3/pipeline/high_priority_pipe HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2015-03-24T13:04:26Z
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: 47281222-f0ff-4f80-9a4d-0140ff77dcd0
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2015-03-24T13:04:26Z/1800/host;x-bce-date/b1ad7075b37616b846a356d1db86e73abba1aed51b4d7b0d95321f69f17250b1
```

请求消息体格式（HTTP Request Body）

音视频转码的API服务要求使用JSON格式的结构体来描述一个请求的具体内容。 作为示例，以下是一个标准的用户创建任务时的请求消息体格式：

```
{
  "targetKey": "output_example.flv",
  "sourceKey": "input_example.mp4",
  "presetName": "bce.video_mp4_1920x1080_3660kbps"
}
```

请求返回格式（HTTP Response）

音视频转码的API服务均采用JSON格式的消息体作为响应返回的格式。 作为示例，以下是一个标准的用户查询队列时的完整的请求返回：

```
HTTP/1.1 200 OK
Transfer-Encoding: chunked
x-bce-request-id: 47281222-f0ff-4f80-9a4d-0140ff77dcd0
Cache-Control: no-cache
Server: BWS
Date: Tue, 24 Mar 2015 13:04:27 GMT
Content-Type: application/json;charset=UTF-8

{
   "pipelineName": "high_priority_pipe",
   "sourceBucket": "examplesourceBucket",
   "targetBucket": "exampletargetBucket",
   "pipelineSize": 5,
   "state":"ACTIVE",
   "lastUpdateTime": "15-03-24T13:02:00Z",
   "description": "Example Pipeline for Demo Purpose.",
   "jobStatus":{
      "total": 0,
      "running": 0,
      "pending": 0,
      "success": 0,
      "failed": 0
   }
}
```

错误信息格式

HTTP状态码	错误码	错误信息	描述
404	NoSuchPreset	The requested preset does not exist. Please double confirm your preserId or the existence of the preset.	所访问的Preset不存在，请重新确定指定的presetName是否正确或确认指定Preset是否存在。
404	NoSuchPipeline	The requested pipeline does not exist. Please confirm your pipelineName or the existence of the pipeline.	所访问的Pipeline不存在，请重新确定指定的pipelineName是否正确或确认指定Pipeline是否存在。
404	NoSuchJob	The requested job does not exist. Please confirm your jobId or the existence of the job.	所访问的Job不存在，请重新确定指定的jobId是否正确或确认指定Job是否存在。
400	PipelineFull	The requested pipeline is full. Please wait for the pipeline to fullfill more jobs or have a new one.	任务队列已满，请等待队列中的任务完成或者创建一个新的队列。
400	InvalidBucket	You have no access to bucket: {bucket name}. Please confirm your bucket name or the existence of the bucket.	用户指定的Bucket无法访问，请确认Bucket的拼写或确认Bucket是否存在。
400	PipelineNotEmpty	Pipeline have jobs in pending or running status. Please wait for the jobs to be completed and try again.	队列中尚包含等待或运行中的任务，无法继续下一步操作，请等待队列中的任务完成。
404	PipelineNotActive	Pipeline has been deleted. Please confirm your pipelineName or specify another pipeline.	队列已被标记删除，请重新确认pipelineName或者指定其他的Pipeline.
400	PresetNotEmpty	Preset has jobs applied in pending or running status. Please wait for the jobs to be completed and try again.	使用该模板任务尚有处在在等待或运行中的状态，请等待使用该模板的任务全部完成。
400	InvalidParameterException	There are field(s) invalied in the request messsage body, please check.	请求的结构体中包含一个或者多个字段错误，请检查确认。
400	InvalidDelogo	Delogo (去水印) is not supported if preset is transmux	transmux模式下不可设置去水印

通知接口

通知功能可以在音视频转码任务状态转换时主动向开发者服务器推送消息。

创建通知

请求（Request）

- 请求语法：

```
 POST /v{version}/notification HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

- 请求头域：无特殊Header参数

- 请求参数（以下均为Requestbody参数）：

字段名称	字段类型	必要性	字段描述	可选值	默认值
name	String	必选	通知名称。小写字母开头，其余部分由小写字母、下划线 (_) 或数字组成，最多不超过40个字符	-	-
endpoint	String	必选	通知消息接收地址	-	-
type	String	可选	通知消息类型。NONE表示普通通知回调；SIGN表示鉴权模式，会在header中增加Notification-Auth-Expire、Notification-Auth-User和Notification-Auth-Token用于验证。用户接收到回调后可以使用SHA256-HEX传入自己的token以及"POST;endpoint;content;expireTime;user"生成Notification-Auth-Token，与header中的对比进行验证	NONE,SIGN	NONE
token	String	可选	通知消息鉴权token	-	-

- 请求示例：

```
 POST /v3/notification HTTP/1.1
content-length: 87
accept-encoding: gzip, deflate
x-bce-date: 2015-07-03T09:28:13Z
connection: keep-alive
accept: */*
user-agent: python-requests/2.4.0 CPython/2.7.9 Darwin/14.3.0
host: media.bj.baidubce.com
x-bce-request-id: 8776558c-81d9-4f97-8e2c-f977a286095d
content-type: application/json
authorization: bce-auth-v1/e8e4a9ced6794355a9a1b8a20b58d37b/2015-07-03T09:28:13Z/1800/content-type;host;x-bce-date/4a1692dc4bab84f5801f79ea0c1fece3601cf73ecd94409d2a94b3942b971715

{
  "name": "mct_notification",
  "endpoint": "http://mct.notificaitonDomain.com/"
}
```

响应（Reponse）

- 响应头域：无特殊Header参数
- 响应参数：无
- 响应示例：

```
 HTTP/1.1 200 OK
x-bce-request-id: 8776558c-81d9-4f97-8e2c-f977a286095d
Date: Fri, 03 Jul 2015 09:28:13 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Cache-Control: no-cache
```

接收通知回调

- 通知消息：

字段名称	字段类型	字段描述
messageId	String	通知Id

messageId	String	通知Id
messageBody	Object	通知消息体
+ jobId	String	jobid
+ pipelineName	String	队列名
+ jobStatus	String	job状态（SUCCESS, FAILED）
+ createTime	Date	job创建时间
+ startTime	Date	job开始时间
+ endTime	Date	job结束时间
+ error	Object	错误信息
++ code	String	错误码
++ message	String	错误信息
+ source	Object	输入的原始信息
++ sourceKey	String	原始文件的BOS Key
++ clips	Array	待合并的原始视频信息
+++ bucket	String	原始文件的BOS Bucket
+++ sourceKey	String	原始文件的BOS Key
+++ asMasterClip	Boolean	是否指定该片段作为主分片
+++ enableLogo	Boolean	是否允许在该片段添加水印
+++ enableDelogo	Boolean	是否允许在该片段进行去水印
+++ enableCrop	Boolean	是否允许在该片段添进行去黑边
+++ startTimeInSecond	Number	视频片段的起始时间
+++ durationInSecond	Number	视频片段的持续时间
+++ startTimeInMillisecond	Number	视频片段的起始时间（单位ms，与startTimeInSecond同时设置时，优先生效）
+++ durationInMillisecond	Number	视频片段的持续时间（单位ms，与durationInSecond同时设置时，优先生效）
+ target	Object	目前信息
++ targetKey	String	目标文件的BOS key
++ presetName	String	输出处理的模板的presetName
+ output	Object	输出信息
++ Video	Object	视频
+++ durationInSeconds	Number	视频持续时间
+++ sizeInKiloByte	Double	
+++ widthInPixel	Number	宽
+++ heightInPixel	Number	高
+++ frameRate	Number	帧率
+++ mp4MoovSize	Number	
++ Audio	Object	音频
+++ sampleRateInHz	Number	音频采样率
+++ channels	Number	声道
+ secretKeyUserId	String	
+ clusterName	String	视频源信息

notification	String	通知名称
server	String	发送通知的服务host
subscriptionName	String	（兼容QNS通知）
version	String	（兼容QNS通知）
signature	String	（兼容QNS通知）

- 示例：

{ "messageBody": { "jobId": "job-jmkmdp90t4rvkhp0", "pipelineName": "test_notification", "jobStatus": "SUCCESS", "createTime": "2019-12-11T03:09:35Z", "startTime": "2019-12-11T03:09:58Z", "endTime": "2019-12-11T03:10:15Z", "source": { "clips": [{ "bucket": "tescae", "sourceKey": "input/chengdu.mp4" }] }, "target": { "targetKey": "test_notification.mp4", "presetName": "bce.video_mp4_1920x1080_3660kbps" }, "output": { "video": { "durationInSeconds": 15, "sizeInKiloByte": 5804.772461, "widthInPixel": 606, "heightInPixel": 1080, "frameRate": 30, "mp4MoovSize": 9728 }, "audio": { "sampleRateInHz": 44100, "channels": 2 }, "bitRateInKBps": "3095.88" } }, "messageId": "943520a5-14c4-4980-84a0-9c44b8dfe517", "notification": "test_to_dev_8009", "server": "multimedia.bce-testinternal.baidu.com", "subscriptionName": "", "version": "", "signature": "" }

查询通知

请求（Request）

- 请求语法：

```
GET /v{version}/notification/{name} HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

- 请求头域：无特殊Header参数
- 请求参数：无
- 请求示例：

```
GET /v3/notification/mct_notification HTTP/1.1
content-length: 0
accept-encoding: gzip, deflate
x-bce-date: 2015-07-03T09:28:13Z
connection: keep-alive
accept: */*
user-agent: python-requests/2.4.0 CPython/2.7.9 Darwin/14.3.0
host: media.bj.baidubce.com
x-bce-request-id: 8776558c-81d9-4f97-8e2c-f977a286095d
content-type: application/json
authorization: bce-auth-v1/e8e4a9ced6794355a9a1b8a20b58d37b/2015-07-03T09:28:13Z/1800/content-type;host;x-bce-date/4a1692dc4bab84f5801f79ea0c1fece3601cf73ecd94409d2a94b3942b971715
```

响应（Reponse）

- 响应头域：无特殊Header参数
- 响应参数：

字段名称	字段类型	字段描述
name	String	通知名称
endpoint	String	通知消息接收地址

- 响应示例：

```
 HTTP/1.1 200 OK
x-bce-request-id: 8776558c-81d9-4f97-8e2c-f977a286095d
Date: Fri, 03 Jul 2015 09:28:13 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Cache-Control: no-cache

{
  "name" : "mct_notification",
  "endpoint" : "http://mct.notificaitonDomain.com/"
}
```

删除通知

请求（Request）

- 请求语法：

```
 DELETE /v{version}/notification/{name} HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

- 请求头域：无特殊Header参数
- 请求参数：无
- 请求示例：

```
 DELETE /v3/notification/mct_notification HTTP/1.1
content-length: 0
accept-encoding: gzip, deflate
x-bce-date: 2015-07-03T09:28:13Z
connection: keep-alive
accept: */*
user-agent: python-requests/2.4.0 CPython/2.7.9 Darwin/14.3.0
host: media.bj.baidubce.com
x-bce-request-id: 8776558c-81d9-4f97-8e2c-f977a286095d
content-type: application/json
authorization: bce-auth-v1/e8e4a9ced6794355a9a1b8a20b58d37b/2015-07-03T09:28:13Z/1800/content-
type;host;x-bce-date/4a1692dc4bab84f5801f79ea0c1fece3601cf73ecd94409d2a94b3942b971715
```

响应（Reponse）

- 响应头域：无特殊Header参数
- 响应参数：无

- 响应示例：

```
HTTP/1.1 200 OK
x-bce-request-id: 8776558c-81d9-4f97-8e2c-f977a286095d
Date: Fri, 03 Jul 2015 09:28:13 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Cache-Control: no-cache
```

通知列表

请求（Request）

- 请求语法：

```
GET /v{version}/notification HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

- 请求头域：无特殊Header参数
- 请求参数：无
- 请求示例：

```
GET /v3/notification HTTP/1.1
content-length: 0
accept-encoding: gzip, deflate
x-bce-date: 2015-07-03T09:28:13Z
connection: keep-alive
accept: */*
user-agent: python-requests/2.4.0 CPython/2.7.9 Darwin/14.3.0
host: media.bj.baidubce.com
x-bce-request-id: 8776558c-81d9-4f97-8e2c-f977a286095d
content-type: application/json
authorization: bce-auth-v1/e8e4a9ced6794355a9a1b8a20b58d37b/2015-07-03T09:28:13Z/1800/content-type;host;x-bce-date/4a1692dc4bab84f5801f79ea0c1fece3601cf73ecd94409d2a94b3942b971715
```

响应（Reponse）

- 响应头域：无特殊Header参数
- 响应参数：

字段名称	字段类型	字段描述
notifications	Object	通知列表
+ name	String	通知名称
+ endpoint	String	通知消息接收地址

- 响应示例：

```
 HTTP/1.1 200 OK
x-bce-request-id: 8776558c-81d9-4f97-8e2c-f977a286095d
Date: Fri, 03 Jul 2015 09:28:13 GMT
Transfer-Encoding: chunked
Content-Type: application/json;charset=UTF-8
Cache-Control: no-cache

{
  "notifications" : [ {
    "name" : "mct_notification",
    "endpoint" : "http://mct.notificaitonDomain.com/"
  }, {
    "name" : "my_notification",
    "endpoint" : "http://my.notificaitonDomain.com/"
  } ]
}
```

队列接口

创建队列

接口描述

基本接口，用户向服务请求创建任务队列。

请求（Request）

- 请求语法：

```
 POST /v{version}/pipeline HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
connection: keep-alive
accept: */*
host: media.bj.baidubce.com
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

- 请求头域：无特殊Header参数
- 请求参数：

字段名	字段类型	必要性	字段描述	可选值	参数位置
pipelineName	String	必选	队列名称，允许小写字母、数字以及下划线且必须以字母开头，长度小于40个字符	-	requestbody参数
description	String	可选	队列描述，长度小于128个字符	-	requestbody参数
sourceBucket	String	必选	输入Bucket	-	requestbody参数
targetBucket	String	必选	输出Bucket	-	requestbody参数
config	Object	可选	队列的配置	-	requestbody参数
+ capacity	Number	可选	队列的并发能力(默认20)	大于等于1	requestbody参数
+ notification	String	可选	通知名称	-	requestbody参数
+ pipelineType	String	可选	队列类型: 默认 normal，极速转码 acceleration	-	requestbody参数

- 请求示例：

```
 POST /v3/pipeline HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2015-03-24T13:02:00Z
connection: keep-alive
accept: */*
host: media.bj.baidubce.com
x-bce-request-id: 73c4e74c-3101-4a00-bf44-fe246959c05e
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2015-03-24T13:02:00Z/1800/host;x-bce-date/994014d96b0eb26578e039fa053a4f9003425da4bfedf33f4790882fb4c549
{
    "sourceBucket": "exampleluputBucket",
    "targetBucket": "exampletargetBucket",
    "pipelineName": "medium_priority_pipe",
    "description": "The pipeline holding medium priority jobs.",
    "config": {
       "capacity": "5",
       "notification" : "mct_notification",
       "pipelineType": "normal"
    }
}
```

响应（Response）

- 响应头域：无特殊Header参数
- 响应参数：无
- 响应示例：

```
 HTTP/1.1 200 OK
x-bce-request-id: 73c4e74c-3101-4a00-bf44-fe246959c05e
Cache-Control: no-cache
Server: BWS
Date: Tue, 24 Mar 2015 13:02:01 GMT
Content-Type: application/json;charset=UTF-8
```

查询指定队列

接口描述

基本接口，用户通过指定name来查询该指定队列的信息。

请求（Request）

- 请求语法：

```
 http
GET /v{version}/pipeline/{pipelineName} HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

- 请求头域：无特殊Header参数
- 请求参数：无
- 请求示例：

```
 http
GET /v3/pipeline/high_priority_pipe HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2015-03-24T13:04:26Z
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: 4cd11be3-fcaf-4ce5-aa4a-135f3c27b12b
content-type: application/json
authorization: bce-auth-v1/02296dd93f1940a39913d9a406332486/2015-03-24T13:04:26Z/1800/host;x-bce-date/0ffab05c5001f9d80c8d2630561ff2144cf070d1fe838133412c8245615046f9
```

响应（Response）

- 响应头域：无特殊Header参数
- 响应参数

字段名称	字段类型	字段描述
pipelineName	String	用户指定的队列名称，允许小写字母、数字以及下划线且必须以字母开头
sourceBucket	String	用户指定的输入Bucket
targetBucket	String	用户指定的输出Bucket
config	Object	队列配置的对象
+ capacity	Number	队列的并发能力
+ notification	String	通知名称
state	String	队列状态，分为ACTIVE/INACTIVE
createTime	String	创建时间，UTC格式
description	String	用户指定的队列描述
jobStatus	Object	队列中任务的状态集合
+ total	Number	队列中的任务总数
+ running	Number	队列中运行中的任务总数
+ pending	Number	队列中排队中的任务总数
+ success	Number	队列中已执行成功的任务总数
+ failed	Number	队列中执行失败的任务总数

- 响应示例：

```
 http
HTTP/1.1 200 OK
Transfer-Encoding: chunked
x-bce-request-id: 47281222-f0ff-4f80-9a4d-0140ff77dcd0
Cache-Control: no-cache
Server: BWS
Date: Tue, 24 Mar 2015 13:04:27 GMT
Content-Type: application/json;charset=UTF-8

{
    "name": "high_priority_pipe",
    "sourceBucket": "exampleSourceBucket",
    "targetBucket": "exampleTargetBucket",
    "config": {
      "capacity": 5,
      "notification": "mct_notification"
    },
    "state":"ACTIVE",
    "createTime":"2015-03-24T13:04:26Z",
    "description":"A pipleine hoding high priority jobs.",
    "jobStatus":
    {
        "total": 10,
        "running": 2,
        "pending": 1,
        "success": 7,
        "failed": 1
    }
}
```

查询用户所有队列

接口描述

易用性接口，帮助用户查询所拥有的所有队列的详细信息。

请求（Request）

- 请求语法：

```
 http
GET /v{version}/pipeline HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

- 请求头域：无特殊Header参数
- 请求参数：无
- 请求示例：

```
 http
GET /v3/pipeline HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2015-03-24T13:03:20Z
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: cc7500cb-ebc5-4ac2-9018-4befad8e2479
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2015-03-24T13:03:20Z/1800/host;x-bce-
date/c5e36c982e5037841925c6729f1e74df53ebb134745d3e6da2722f2333d390a2
```

响应（Response）

- 响应头域：无特殊Header参数
- 响应参数：

字段名	字段类型	字段描述
pipelines	Object	用户队列的集合
+ pipelineName	String	用户指定的队列名称
+ sourceBucket	String	用户指定的输入Bucket
+ targetBucket	String	用户指定的输出Bucket
+ config	Object	队列配置
++ capacity	Number	队列的并发能力
++ notification	String	通知名称
+ state	String	队列状态，分为ACTIVE/INACTIVE
+ createTime	String	创建时间，UTC格式
+ description	String	用户指定的队列描述
+ status	Object	队列中任务的状态集合
++ total	Number	队列中的任务总数
++ running	Number	队列中运行中的任务总数
++ pending	Number	队列中排队中的任务总数
++ success	Number	队列中已执行成功的任务总数
++ failed	Number	队列中执行失败的任务总数

- 响应示例：

```
 http
HTTP/1.1 200 OK
Transfer-Encoding: chunked
x-bce-request-id: cc7500cb-ebc5-4ac2-9018-4befad8e2479
Cache-Control: no-cache
Server: BWS
Date: Tue, 24 Mar 2015 13:03:21 GMT
Content-Type: application/json;charset=UTF-8

{
    "pipelines": [
        {
            "pipelineName": "high_priority_pipe"
```

```
            "pipelineName": "high_priority_pipe",
            "sourceBucket": "sampleSourceBucket00",
            "targetBucket": "sampleTargetBucket00",
            "config": {
                "capacity":20,
                "notification": "mct_notification"
            },
            "state": "ACTIVE",
            "createTime": "2015-03-24T13:03:20Z",
            "description": "The pipeline holding high priority jobs.",
            "jobStatus": {
                "total": 0,
                "running": 0,
                "pending": 0,
                "success": 0,
                "failed": 0
            }
        },
        {
            "pipelineName": "medium_priority_pipe",
            "sourceBucket": "sampleSourceBucket01",
            "targetBucket": "sampleTargetBucket01",
            "config": {
                "capacity":20
            },
            "state": "ACTIVE",
            "createTime": "2015-03-24T13:03:20Z",
            "description": "The pipeline holding medium priority jobs.",
            "jobStatus": {
                "total": 0,
                "running": 0,
                "pending": 0,
                "success": 0,
                "failed": 0
            }
        },
        {
            "pipelineName": "low_priority_pipe",
            "sourceBucket": "sampleSourceBucket02",
            "targetBucket": "sampleTargetBucket02",
            "config": {
                "capacity":20
            },
            "state": "ACTIVE",
            "createTime": "2015-03-24T13:03:20Z",
            "description": "The pipeline holding low priority jobs.",
            "jobStatus": {
                "total": 0,
                "running": 0,
                "pending": 0,
                "success": 0,
                "failed": 0
            }
        }
    ]
}
```


删除指定队列

接口描述

基本接口，用户向服务请求删除指定pipelineName的任务队列。

请求（Request）

- 请求语法：

```
 http
DELETE /v{version}/pipeline/{pipelineName} HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
connection: keep-alive
accept: */*
host: media.bj.baidubce.com
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

- 请求头域：无特殊Header参数
- 请求参数：无
- 请求示例：

```
 http
DELETE /v3/pipeline/high_priority_pipe HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2015-03-24T13:06:02Z
connection: keep-alive
accept: */*
host: media.bj.baidubce.com
x-bce-request-id: 6d0b0a36-2ffe-49d4-9d81-333a9ab9417e
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2015-03-24T13:06:02Z/1800/host;x-bce-
date/02f64774999996903cffa5ae4d6eef436127a96f581a4e8467497e239d824be8
```

响应（Response）

- 响应头域：无特殊Header参数
- 响应参数：无
- 响应示例：

```
 http
HTTP/1.1 200 OK
x-bce-request-id: 6d0b0a36-2ffe-49d4-9d81-333a9ab9417e
Cache-Control: no-cache
```

更新指定队列

接口描述

基本接口，用户向服务请求更新任务队列。

请求（Request）

- 请求语法：

```
PUT /v{version}/pipeline/{pipelineName} HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
connection: keep-alive
accept: */*
host: media.bj.baidubce.com
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

- 请求头域：无特殊Header参数
- 请求参数：

字段名	字段类型	必要性	字段描述	可选值	参数位置
pipelineName	String	必选	要更新队列名称	-	requestbody参数
description	String	可选	队列描述，长度小于128个字符	-	requestbody参数
sourceBucket	String	必选	输入Bucket	-	requestbody参数
targetBucket	String	必选	输出Bucket	-	requestbody参数
config	Object	可选	队列的配置	-	requestbody参数
+ capacity	Number	可选	队列的并发能力(默认20)	大于等于1	requestbody参数
+ notification	String	可选	通知名称	-	requestbody参数

- 请求示例：

```
PUT /v3/pipeline/medium_priority_pipe HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2020-02-14T14:59:24Z
connection: keep-alive
accept: */*
host: media.bj.baidubce.com
x-bce-request-id: deda676c-0892-4e14-9881-9d6f842d5728
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2020-02-14T14:59:24Z/1800/host;x-bce-
date/994014d96b0eb26578e039fa053a4f9003425da4bfedf33f4790882fb4c549
{
    "sourceBucket": "exampleluputBucket",
    "targetBucket": "exampletargetBucket",
    "pipelineName": "medium_priority_pipe",
    "description": "The pipeline holding medium priority jobs.",
    "config": {
        "capacity": "10",
        "notification" : "mct_notification"
    }
}
```

响应（Response）

- 响应头域：无特殊Header参数
- 响应参数：无
- 响应示例：

```
HTTP/1.1 200 OK
x-bce-request-id: deda676c-0892-4e14-9881-9d6f842d5728
Cache-Control: no-cache
Server: BWS
Date: Fri, 14 Feb 2020 06:59:24 GMT
Content-Type: application/json;charset=UTF-8
```

媒体信息获取接口

查询指定媒体信息

接口描述

用户通过Bucket+BOS Key的URL Encoded的结果获取指定音视频文件的媒体信息。

请求（Request）

- 请求语法：

```
GET /v{version}/mediainfo?bucket={BOS Bucket}&key={uriComponentEncode(BOS Key)} HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

- 请求头域：无特殊Header参数
- 请求参数：无
- 请求示例：

```
GET /v3/mediainfo?bucket=samplebucket&key=sampleforderpath%2Fsampleoutput.mp4 HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2015-03-24T13:37:10Z
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: 3807ce30-5264-45f2-9b52-26b78e24a750
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2015-03-24T13:37:10Z/1800/host;x-bce-
date/3e1bf9f50ae1fca2d704d61567810dde946fff3ca2e455676455a6f5c8cce596
```

响应（Reponse）

- 响应头域：无特殊Header参数
- 响应参数：

字段名称	字段类型	字段描述
bucket	String	音视频文件所在的BOS的Bucket
key	String	音视频文件的BOS的Key
fileSizeInByte	Number	音视频文件的大小
durationInSecond	Number	音视频媒体时长
container	String	音视频文件的容器类型
etag	String	文件的版本标识（详见BOS接口公共返回头）
type	String	文件类型
video	Object	视频信息集合
+ codec	String	视频文件的编码规格
+ heightInPixel	Number	视频高度
+ widthInPixel	Number	视频宽度
+ bitRateInBps	Number	视频媒体的码率
+ frameRate	Number	视频媒体的帧率
+ rotate	Number	旋转角度（部分视频包含该参数）
+ dar	String	视频显示宽高比，如 "16:9" （部分视频包含该参数）
audio	Object	音频信息集合
+ codec	String	音频文件的编码规格
+ channels	Number	音频文件的声道信息
+ sampleRateInHz	Number	音频文件的采样率
+ bitRateInBps	Number	音频文件的码率

- 响应示例：

```
HTTP/1.1 200 OK
Transfer-Encoding: chunked
x-bce-request-id: 3807ce30-5264-45f2-9b52-26b78e24a750
Cache-Control: no-cache
Server: BWS
Date: Tue, 24 Mar 2015 13:37:10 GMT
Content-Type: application/json;charset=UTF-8

{
    "bucket": "samplebucket",
    "key": "samplefolderpath/sampleoutput.mp4",
    "fileSize": 102400,
    "durationInSecond": 60,
    "container": "mp4",
    "eTag": "bf407c4ca0dc4f2f7d581ec94ca36876",
    "type": "video",
    "video": {
        "codec": "h264",
        "heightInPixel": 1024,
        "widthInPixel": 768,
        "bitRateInBps": 2500
    },
    "audio": {
        "codec": "acc",
        "channels": 2,
        "sampleRateInHz": 96000,
        "bitRateInBps": 1100
    }
}
```

明水印接口

创建水印

请求（Request）

- 请求语法：

```
POST /v{version}/watermark HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

- 请求头域：无特殊Header参数
- 请求参数（以下均为Requestbody参数）：

字段名称	字段类型	必要性	字段描述	可选值	默认值
bucket	String	必选	BOS存储上水印文件Bucket	-	-
key	String	必选	BOS存储上水印文件Key，支持JPG、PNG、APNG、BMP、PBM、TIF、GIF 、MOV等格式，其中MOV、GIF、APNG为动态水印	-	-
verticalAlignment	String	可选	垂直对齐方式	top, center, bottom	top
horizontalAlignment	String	可选	水平对齐方式	left, center, right	left
verticalOffsetInPixel	Number	可选	垂直偏移，该参数仅在verticalAlignment设置为top或bottom时有效，单位：像素	0 ~ 3072	0
horizontalOffsetInPixel	Number	可选	水平偏移，该参数仅在horizontalAlignment设置为left或right时有效，单位：像素	0 ~ 4096	0
timeline	Object	可选	水印有效显示起止时间（仅当watermarkId被设置到Preset.watermarks.image多水印参数中时该字段可生效）	-	-
+ startTimeInMillisecond	Number	可选	水印显示起始时间，单位：毫秒	大于等于0	-（为空表示从第0s开始）
+ durationInMillisecond	Number	可选	水印显示持续时间，单位：毫秒	大于等于0	-（为空表示持续视频时长）
repeated	Number	可选	（动态）水印重复显示次数，为0表示无限循环（仅当watermarkId被设置到Preset.watermarks.image多水印参数中时该字段可生效）	大于等于0	1
allowScaling	Bool	可选	是否允许自动进行缩放（仅当watermarkId被设置到Preset.watermarks.image多水印参数中时该字段可生效）	true、false	true

- 请求示例：

```
POST /v3/watermark HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2015-03-24T13:08:44Z
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: d97c57d0-ca44-4d1c-bfeb-941a92440968
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2015-03-24T13:08:44Z/1800/host;x-bce-date/7e21c9cf1e4e2cc6921a407a388fe98df122c53b9f509043d841be76eb09a1f9

{
    "bucket" : "samplebucket",
    "key" : "samplefolderpath/samplewatermark.png",
    "verticalOffsetInPixel" : 0,
    "horizontalOffsetInPixel" : 0
}
```

响应（Reponse）

- 响应头域：无特殊Header参数
- 响应参数：

字段名称	字段类型	字段描述
watermarkId	String	水印的唯一标识

- 响应示例：

```
HTTP/1.1 200 OK
Transfer-Encoding: chunked
x-bce-request-id: d97c57d0-ca44-4d1c-bfeb-941a92440968
Cache-Control: no-cache
Server: BWS
Date: Tue, 24 Mar 2015 13:34:07 GMT
Content-Type: application/json;charset=UTF-8

{
    "watermarkId" : "wmk-lsdspxdastmnbama"
}
```

查询指定水印

请求（Request）

- 请求语法：

```
GET /v{version}/watermark/{watermarkId} HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

- 请求头域：无特殊Header参数

- 请求参数：无
- 请求示例：

```
POST /v3/watermark/wmk-lsdspxdastmnbama HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2015-03-24T13:08:44Z
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: 9c1f8229-a8d4-46e1-b8e2-25412e0eee63
content-type: application/json
authorization: bce-auth-v1/535f3834e538448aa88f3c589bab2ea3/2015-03-24T13:08:44Z/1800/host;x-bce-date/7e21c9cf1e4e2cc6921a407a388fe98df122c53b9f509043d841be76eb09a1f9
```

响应（Reponse）

- 响应头域：无特殊Header参数
- 响应参数：与[创建水印/请求/请求参数]保持一致，增加以下字段

字段名称	字段类型	字段描述
watermarkId	String	水印的唯一标识
createTime	Date	水印创建时间

- 响应示例：

```
HTTP/1.1 200 OK
Transfer-Encoding: chunked
x-bce-request-id: 9c1f8229-a8d4-46e1-b8e2-25412e0eee63
Cache-Control: no-cache
Server: BWS
Date: Tue, 24 Mar 2015 13:34:07 GMT
Content-Type: application/json;charset=UTF-8

{
    "watermarkId" : "wmk-lsdspxdastmnbama",
    "createTime": "2015-05-11T12:42:21Z",
    "bucket" : "samplebucket",
    "key" : "samplefolderpath/samplewatermark.png",
    "verticalOffsetInPixel" : 0,
    "horizontalOffsetInPixel" : 0
}
```

查询当前用户水印

请求（Request）

- 请求语法：

```
GET /v{version}/watermark HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

- 请求头域：无特殊Header参数
- 请求参数：无
- 请求示例：

```
 GET /v3/watermark HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2015-03-24T13:08:44Z
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: 249ac308-4554-4825-ab0f-867dd92024d8
content-type: application/json
authorization: bce-auth-v1/02296dd93f1940a39913d9a406332486/2015-03-24T13:08:44Z/1800/host;x-bce-date/cef8a3207e29c4663292c42665de1154e760c734f82248b36a71be2bb3281b1d
```

响应（Reponse）

- 响应头域：无特殊Header参数
- 响应参数：与[创建水印/请求/请求参数]保持一致，增加以下字段

字段名称	字段类型	字段描述
watermarkId	String	水印的唯一标识
createTime	Date	水印创建时间

- 响应示例：

```
HTTP/1.1 200 OK
Transfer-Encoding: chunked
x-bce-request-id: 249ac308-4554-4825-ab0f-867dd92024d8
Cache-Control: no-cache
Server: BWS
Date: Tue, 24 Mar 2015 13:34:07 GMT
Content-Type: application/json;charset=UTF-8

{
   "watermarks": [
      {
         "watermarkId" : "wmk-lsdspxdastmnbama",
         "createTime": "2015-05-11T12:42:21Z",
         "bucket" : "samplebucket",
         "key" : "samplefolderpath/samplewatermark.png",
         "verticalOffsetInPixel" : 0,
         "horizontalOffsetInPixel" : 0
      },
      {
         "watermarkId" : "wmk-feji7exr57r3824x",
         "createTime": "2015-05-07T12:34:20Z",
         "bucket" : "samplebucket",
         "key" : "samplefolderpath/samplewatermark1.png",
         "verticalOffsetInPixel" : 150,
         "horizontalOffsetInPixel" : 40
      },
      {
         "watermarkId" : "wmk-lsdspxdastmnbama",
         "createTime": "2015-05-08T12:22:15Z",
         "bucket" : "samplebucket2",
         "key" : "samplefolderpath/samplewatermark.png",
         "verticalOffsetInPixel" : 0,
         "horizontalOffsetInPixel" : 200
      }
   ]
}
```

删除水印

请求（Request）

- 请求语法：

```
DELETE /v{version}/watermark/{watermarkId} HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

- 请求头域：无特殊Header参数
- 请求参数：无
- 请求示例：

```
DELETE /v3/watermark/wmk-lsdspxdastmnbama HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2015-03-24T13:08:44Z
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: f922935d-cf8a-4078-a43a-dc3475ab3a70
content-type: application/json
authorization: bce-auth-v1/535f3834e538448aa88f3c589bab2ea3/2015-03-24T13:08:44Z/1800/host;x-bce-date/19468bb170d1073c5cdd292f4424d80d259d0a17528648426f305d39a2d6c452
```

响应（Response）

- 响应头域：无特殊Header参数
- 响应参数：无
- 响应示例：

```
HTTP/1.1 200 OK
x-bce-request-id: f922935d-cf8a-4078-a43a-dc3475ab3a70
Cache-Control: no-cache
```

常见异常

水印不存在

- 异常代码：404
- 异常字段：watermark: XXXX does not exist
- 产生原因： 1.试图查询/删除不存在（未创建/已删除）的水印 2.创建转码模板使用了不存在的水印

水印图片不存在

- 异常代码：404
- 异常字段：bos object: XXXX does not exist
- 产生原因： 1.创建水印时使用了不存在的bos object 2.创建job时引用了不存在的bos object（转码源视频或水印图片）

水印无法删除

- 异常代码：400
- 异常字段：watermark is in use, please delete the relative presets first
- 产生原因： 1.试图删除的水印正在被当前活跃的preset使用，建议用户先删除对应的preset

Transmux模式下增加水印

- 异常代码：400
- watermark is not supported in Transmux mode
- 产生原因： 1.试图在Transmux模式下添加水印，改模式不进行重新编码所以无法添加水印

视频转码模板接口

创建模板

接口描述

当系统预设的Preset无法满足需求时，用户可以通过此接口创建自定义的Preset。

请求（Request）

- 请求语法：

```
POST /v{version}/preset HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
connection: keep-alive
accept: */*
host: media.bj.baidubce.com
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

- 请求头域：无特殊Header参数
- 请求参数（以下均为Requestbody参数）：

字段名称	字段类型	必要性	字段描述	可选值	默认值
presetName	String	必选	模板名称	-	-
description	String	可选	转码模板描述	-	-
container	String	必选	音视频文件的容器	mp4, flv, hls, mp3, m4a, webm, a-hls, pcm, dash, ts	-
transmux	Bool	可选	是否仅执行容器格式转换	true, false	false
clip	Object	可选	是否截取音视频片段	-	-
+ startTimeInSecond	Number	可选	视频片段的起始时间	-	-
+ durationInSecond	Number	可选	视频片段的持续时间	-	-
audio	Object	可选	音频输出信息的集合，不设置audio相关参数则转码输出不包含音频流，如果保留音频流则必须设置此对象	-	-
+ codec	String	可选	音频编码格式	aac, mp3, ac3, pcm, opus	-
+ bitRateInBps	Number	可选	音频目标码率（输出为pcm格式时不可选，设置静音mute时可不填，否则必选）	大于0	-
+ sampleRateIn	Number	可选	音频采样率	22050, 32000, 44100, 48000, 96000	不填写，表示与输

Hz					入保持一致
+ channels	Number	可选	音频声道数目	1, 2	不填写，表示与输入一致
+ pcmFormat	String	可选	PCM音频格式，仅当container=pcm时有效	s16le	-
+ volumeAdjust	Object	可选	音量相关参数设置	-	-
++ mute	Bool	可选	是否进行静音操作，当设置了mute时，Job中不允许有audio类型的insert。mute和其它audio参数同时设置时，优先进行静音（mute）操作	true, false	false
++ norm	Bool	可选	是否进行音频归一化操作	true, false	false
++ gain	Number	可选	音量调节的大小，单位db，值为正则增大音量	-60 ~ 60	-
+ denoise	Object	可选	音频降噪参数设置	-	-
++ strength	Number	可选	音频降噪强度, 值越高降噪能力越强	0.1 ~ 1.0	-
video	Object	可选	视频输出信息的集合，不设置video相关参数则转码输出不包含视频流，如果保留视频流则必须设置此对象	-	-
+ codec	String	可选	视频编码格式	h264, h265(搭配选择profile字段为main), h265_bd265, svt_av1, vp8, vp9	h264
+ codecOptions	Object	可选	视频编码的配置选项	-	-
++ profile	String	可选	档次	baseline, main, high	baseline
++ bFrames	Number	可选	B帧数	0-7	-
++ bPyramid	String	可选	使用B帧作为参考帧（ps：当选择codec为h264才可以设置此参数。）none：禁用； strict：严格分层的金字塔结构； normal：非严格分层的金字塔结构；	none, normal, hieb	normal

			hieb：多层级的参考B帧（当bFrames=7且profile=main/high时才能开启多层级的参考B帧）。		
+ rateControl	String	可选	码率控制方式，crf模式通过调整码率来实现最优质量；相较于vbr，cbr生成的视频码率在目标码率上下浮动范围小，更接近目标码率。感知编码由cae模式控制码率实现。	crf，vbr，cbr，cae	crf
+ codecEnhance	Boolean	必选	ROI增强，是感知编码核心参数：针对人脸区域提升画质、显著性增强。	true，false	false
+ crf	Number	可选	恒定质量因子，若设置了crf值，bitRateInBps可表示最大目标码率	1 ~ 51	-
+ bitRateInBps	Number	必选	视频目标码率	32000bps ~ 500Mbps	-
+ maxRateInBps	Number	可选	视频最大码率 。若选择码率控制方式为crf/cae，需要设置最大码率，在保证视频质量最优的同时，最终码率不超过设置的最大码率上限。	32000bps ~ 500Mbps	-
+ maxFrameRate	Number	可选	目标视频最大帧率	10,15, 23.97, 24, 25, 29.97, 30, 50, 60	-
+ maxWidthInPixel	Number	可选	目标视频的最大宽度	128 ~ 8192	不填写，表示与原始视频保持一致
+ maxHeightInPixel	Number	可选	目标视频的最大高度	96 ~ 7680	不填写，表示与原始视频保持一致
+ sizingPolicy	String	可选	尺寸伸缩策略	keep、shrinkToFit、stretch、shrinkToFitBlur，keep表示保持原始视频宽高比，shrinkToFit表示保持原始视频宽高比并加黑边，stretch表示拉伸原始视频，shrinkToFitBlur同shrinkToFit除了黑边替换为高斯模糊	keep
+ autoAdjustResolution	Bool	可选	当原视频为竖形时，自动调整模板的宽小于高，保证缩放比最小，反之亦然（仅当sizingPolicy为keep时可以设置）	true，false	false

			量）		
+ playbackSpeed	Number	可选	回放速度，值低于1.0时为减速视频，高于1.0时为加速视频（不可同时指定音频设置）	0.05 ~ 20.0	-
encryption	Object	可选	HLS加解密信息的集合	-	-
+ strategy	String	必选	视频加密策略	Fixed: 表示固定密钥加密，使用用户指定的密钥对视频进行加密，此时需要aesKey； Open：开放密钥，系统自动生成加密密钥，密钥公开，不设访问控制； PlayerBinding：绑定播放器，系统自动生成加密密钥，密钥设有访问控制； PlayerBinding模式下密钥设有访问控制，安全性比较高，推荐使用PlayerBinding模式。	-
+ aesKey	String	必选	AES128加密密钥	-	-
watermarkId	String	可选	水印id(当transmux=true时不允许添加水印)	-	-
watermarks	Object	可选	多水印设置，不可同时指定watermarkId和watermarks	-	-
+ image	Array	必选	多水印watermarkId数组	size最大为5	-
digitalWmId	String	可选	需嵌入的数字水印模板ID	需从用户已创建的数字水印模板选择	-
digitalWmSecretKeyId	String	可选	数字水印密钥模板ID，密钥用于对水印加密嵌入，提取水印需提供正确密钥	需从用户已创建的数字水印密钥模板选择，生效优先级高于转码模板中的密钥	-
digitalWmAlgVersion	Integer	可选	算法版本号	0 ~ 2	0
digitalWmStrength	Double	可选	(算法1、2有效)数字水印嵌入强度，对同一算法，强度越高则抗攻击能力越强，隐蔽性越差	0 ~ 1	0.5
transCfg	Object	可选	转码配置信息	-	-
+ transMode	String	必选	转码模式	normal, twopass, cae, cae_enhanced, cae_external, cae_external_sr,cae_external_sr_vis super_resolution，super_resolution_vis。当转码模式为twopass, cae, cae_enhanced时，video不能为空； cae_external_sr cae带主观增强,cae_external_sr_vis cae带主观增强vis模型。super_resolution 超分辨率。 super_resolution_vis vis模型超分辨率	-
extraCfg	Object	可选	转码额外配置/视频处理类		
+ waterm					

arkDisableWhitelist	String	可选	设置不加水印的条件	当前可设置 portrait，表示竖屏视频不加水印	-
+ segmentDurationInSecond	Number	可选	设置分片时长（仅当container为hls,a-hls,dash时可以设置）	取值范围 1.0 ~ 60.0，浮点数精度为小数点后三位以内	-
+ gopLength	Number	可选	设置gop长度（仅当配置了video参数时可以设置；当segmentDurationInSecond和gopLength参数共存时，建议保证segmentDurationInSecond*原视频帧率 为gopLength的整数倍）	[0, 500]，其中0表示结果视频只包含i帧	-
+ skipBlackFrame	Bool	可选	智能检测并裁剪片头黑帧，最长截取前5s黑帧	true，false	false
+ horiToVeri	Bool	可选	视频横转竖	true, false	false
+ stabilization	Bool	可选	视频去抖动	true, false	false
+ autoDesubtitle	Integer	可选	智能去字幕	1: 字幕区域填充背景；2: 裁剪掉字幕区域	NULL
+ autoDelogo	Bool	可选	是否自动去水印。默认手动去水印（位置大小参数需在转码任务参数内配置）	true，false	false
+ delogoMode	String	可选	去水印模式	Normal：高斯模糊；Inpainting：背景填充	Normal
+ colorEnhance	Bool	可选	色彩增强	true，false	false
+ aiVideoEnhance	Bool	可选	视频细节增强,不改变分辨率	true，false	false
+ enhanceStrength	Float	可选	细节增强强度，越大越锐利	0-1	1
+					

faceEnhanceModel	String	可选	人脸增强	根据人脸增强时的细节生成水平，可选strong_generative，ultra_generative	-
+ aiSdrToHdr	Bool	可选	智能HDR	true，false；ps：智能hdr必须选择h265编码，main10编码规格	false
+ superResolution	Bool	可选	超分辨率，ps：最多可超分3-5倍	true，false	false
+ superResolutionVersion	Integer	可选	超分模型选择	-1：仅缩放 ；0: 模型0，速度较快，适合互联网UGC内容；1：模型1，速度慢，适合影视剧；2：模型2，速度慢，适合较老的影视剧；3：模型3，速度快，适合UGC和影视剧，细节生成能力较弱；4：模型4，速度快，适合UGC和影视剧，细节生成能力强	0
+ frameInterpolate	Bool	可选	智能插帧	true, false	false
+ aiVideoScratchRemove	Float	可选	老片修复_去划痕	0～1 数字越大，去划痕灵敏度越高	-
+ aiVideoDenoise	Float	可选	老片修复_去噪	0～1 数字越大，去噪强度越大	-
+ aiVideoColorization	Bool	可选	老片修复_上色	true，false	false
+ preserveMetadata	Bool	可选	保留源视频的metadata	true, false	false

- 请求示例：

```
POST /v3/preset HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2015-03-24T13:08:44Z
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: 3807ce30-5264-45f2-9b52-26b78e24a750
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2015-03-24T13:08:44Z/1800/host;x-bce-date/7e21c9cf1e4e2cc6921a407a388fe98df122c53b9f509043d841be76eb09a1f9

{
    "presetName": "customlized_preset",
    "description": "A example preset description",
    "container": "mp4",
    "clip": {
        "startTimeInSecond": 0,
        "durationInSecond": 60
    },
    "audio": {
        "bitRateInBps": 256000,
            "volumeAdjust": {
            "norm": true,
            "gain": 10,
        }
    },
    "video": {
        "codec": "h264",
        "codecOptions": {
            "profile": "baseline"
        },
        "bitRateInBps": 1024000,
        "crf": 20,
        "maxFrameRate": 30,
        "maxWidthInPixel": 4096,
        "maxHeightInPixel": 3072,
        "sizingPolicy": "keep",
        "autoAdjustResolution": true,
        "playbackSpeed": 1.5
    },
    "watermarks": {
"image": ["wmk-idkmvixzaia83bem","wmk-hkjws5t44gunyz8m"]
}
}
```

响应（Response）

- 响应头域：无特殊Header参数
- 响应参数：无
- 响应示例：

```
HTTP/1.1 200 OK
Transfer-Encoding: chunked
x-bce-request-id: 3807ce30-5264-45f2-9b52-26b78e24a750
Cache-Control: no-cache
Server: BWS
    Date: Tue, 24 Mar 2015 13:37:10 GMT
Content-Type: application/json;charset=UTF-8
```

查询指定模板

接口描述

通过presetName查询指定的模板信息。

请求（Request）

- 请求语法：

```
GET /v{version}/preset/{presetId} HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

- 请求头域：无特殊Header参数
- 请求参数：无
- 请求示例：

```
GET /v3/preset/bce.video_mp4_1920x1080_3660kbps HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2015-03-24T13:37:10Z
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: 3807ce30-5264-45f2-9b52-26b78e24a750
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2015-03-24T13:37:10Z/1800/host;x-bce-date/3e1bf9f50ae1fca2d704d61567810dde946fff3ca2e455676455a6f5c8cce596
```

响应（Response）

- 响应头域：无特殊Header参数
- 响应参数：与[创建模板/请求/请求参数]保持一致，增加以下字段

字段名称	字段类型	字段描述
state	String	模板状态，ACTIVE/INACTIVE
createdTime	String	模板创建的UTC格式的时间

- 响应示例：

```
 HTTP/1.1 200 OK
Transfer-Encoding: chunked
x-bce-request-id: 3807ce30-5264-45f2-9b52-26b78e24a750
Cache-Control: no-cache
Server: BWS
Date: Tue, 24 Mar 2015 13:37:10 GMT
Content-Type: application/json;charset=UTF-8

{
    "state": "ACTIVE",
    "createdTime": "2015-03-24T13:34:07Z",
    "presetName": "bce.video_mp4_1920x1080_3660kbps",
    "description": "A example preset description",
    "container": "mp4",
    "clip": {
        "startTimeInSecond": 0,
        "durationInSecond": 60
    },
    "audio": {
        "bitRateInBps": 256000,
        "channels": "auto"
    },
    "video": {
        "codec": "h264",
    "codecOptions": {
            "profile": "baseline"
    },
    "bitRateInBps": 3660000,
    "maxFrameRate": 15,
    "maxWidth": 1920,
    "maxHeight": 1080,
    "sizingPolicy": "keep",
    "autoAdjustResolution": true,
    "playbackSpeed": 1.5
    },
    "watermarkId": {
"image": ["wmk-idkmvixzaia83bem","wmk-hkjws5t44gunyz8m"]
    }
}
```

查询当前用户模板及所有系统模板

接口描述

用户查询其名下及系统提供的所有的模板，具体有哪些系统模板可以参考系统内置模板。

请求（Request）

- 请求语法：

```
 GET /v{version}/preset HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
connection: keep-alive
accept: */*
host: media.bj.baidubce.com
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

- 请求头域：无特殊Header参数

- 请求参数：无
- 请求示例：

```
GET /v3/preset HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2015-03-24T13:37:10Z
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: 3807ce30-5264-45f2-9b52-26b78e24a750
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2015-03-24T13:37:10Z/1800/host;x-bce-date/3e1bf9f50ae1fca2d704d61567810dde946fff3ca2e455676455a6f5c8cce596
```

响应（Response）

- 响应头域：无特殊Header参数
- 响应参数：与[创建模板/请求/请求参数]保持一致，增加以下字段

字段名称	字段类型	字段描述
state	String	模板状态，ACTIVE/INACTIVE
createdTime	String	模板创建的UTC格式的时间

- 响应示例：

```
HTTP/1.1 200 OK
Transfer-Encoding: chunked
x-bce-request-id: 3807ce30-5264-45f2-9b52-26b78e24a750
Cache-Control: no-cache
Server: BWS
Date: Tue, 24 Mar 2015 13:37:10 GMT
Content-Type: application/json;charset=UTF-8

{
  "presets": [
    {
      "presetName": "bce.audio_hls_128kbps",
      "description": "hls audio，128kbps",
      "container": "hls",
      "transmux": false,
      "clip": {
        "startTimeInSecond": 0
      },
      "audio": {
        "sampleRateInHz": 44100,
        "channels": 2
      },
      "createTime": "2015-04-07T12:01:08Z"
    },
    {
      "presetName": "bce.audio_hls_160kbps",
      "description": "hls audio，160kbps",
      "container": "hls",
      "transmux": false,
      "clip": {
        "startTimeInSecond": 0
      },
      "audio": {
```

```
      "sampleRateInHz": 44100,
      "channels": 2
    },
    "createTime": "2015-04-07T12:01:08Z"
  },
  {
    "presetName": "bce.audio_hls_64kbps",
    "description": "hls audio，64kbps",
    "container": "hls",
    "transmux": false,
    "clip": {
      "startTimeInSecond": 0
    },
    "audio": {
      "sampleRateInHz": 22050,
      "channels": 2
    },
    "createTime": "2015-04-07T12:01:08Z"
  },
  {
    "presetName": "bce.audio_mp3_128kbps",
    "description": "mp3 audio，128kbps",
    "container": "mp3",
    "transmux": false,
    "clip": {
      "startTimeInSecond": 0
    },
    "audio": {
      "sampleRateInHz": 44100,
      "channels": 2
    },
    "createTime": "2015-04-07T12:01:09Z"
  },
  {
    "presetName": "bce.audio_mp3_160kbps",
    "description": "mp3 audio，160kbps",
    "container": "mp3",
    "transmux": false,
    "clip": {
      "startTimeInSecond": 0
    },
    "audio": {
      "sampleRateInHz": 44100,
      "channels": 2
    },
    "createTime": "2015-04-07T12:01:09Z"
  },
  {
    "presetName": "bce.audio_mp3_192kbps",
    "description": "mp3 audio，192kbps",
    "container": "mp3",
    "transmux": false,
    "clip": {
      "startTimeInSecond": 0
    },
    "audio": {
      "sampleRateInHz": 44100,
      "channels": 2
    },
    "createTime": "2015-04-07T12:01:09Z"
  },
  {
```

```
      "presetName": "bce.audio_mp3_320kbps",
      "description": "mp3 audio，320kbps",
      "container": "mp3",
      "transmux": false,
      "clip": {
        "startTimeInSecond": 0
      },
      "audio": {
        "sampleRateInHz": 44100,
        "channels": 2
      },
      "createTime": "2015-04-07T12:01:08Z"
    },
    {
      "presetName": "bce.audio_mp3_64kbps",
      "description": "mp3 audio，64kbps",
      "container": "mp3",
      "transmux": false,
      "clip": {
        "startTimeInSecond": 0
      },
      "audio": {
        "sampleRateInHz": 22050,
        "channels": 2
      },
      "createTime": "2015-04-07T12:01:09Z"
    }
  ]
}
```


删除指定模板

接口描述

用于删除用户指定presetName的用户模板

请求（Request）

- 请求语法：

```
DELETE /v{version}/preset/{presetName} HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
connection: keep-alive
accept: */*
host: media.bj.baidubce.com
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

- 请求头域：无特殊Header参数
- 请求参数：无
- 请求示例：

```
DELETE /v3/preset/user-video-mp4-1080p-3660kbps HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2015-03-24T13:06:02Z
connection: keep-alive
accept: */*
host: media.bj.baidubce.com
x-bce-request-id: 6d0b0a36-2ffe-49d4-9d81-333a9ab9417e
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2015-03-24T13:06:02Z/1800/host;x-bce-date/02f64774999996903cffa5ae4d6eef436127a96f581a4e8467497e239d824be8
```

响应（Response）

- 响应头域：无特殊Header参数
- 响应参数：无
- 响应示例：

```
HTTP/1.1 200 OK
x-bce-request-id: 6d0b0a36-2ffe-49d4-9d81-333a9ab9417e
Cache-Control: no-cache
```

更新指定模板

接口描述

用户可以通过此接口更新指定Preset。

请求（Request）

- 请求语法：

```
PUT /v{version}/preset/{presetName} HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
connection: keep-alive
accept: */*
host: media.bj.baidubce.com
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

- 请求头域：无特殊Header参数
- 请求参数：同创建模板请求
- 请求示例：

```
PUT /v3/preset/customlized_preset HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2020-02-14T15:12:37Z
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: 0969b91e-1890-470b-b93b-97939f672e54
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2020-02-14T15:12:37Z/1800/host;x-bce-date/7e21c9cf1e4e2cc6921a407a388fe98df122c53b9f509043d841be76eb09a1f9

{
    "presetName": "customlized_preset",
    "description": "A example preset description",
    "container": "mp4",
    "clip": {
        "startTimeInSecond": 0,
        "durationInSecond": 60
    },
    "audio": {
        "bitRateInBps": 256000
    },
    "video": {
        "codec": "h264",
        "codecOptions": {
            "profile": "baseline"
        },
        "bitRateInBps": 1024000,
        "maxFrameRate": 30,
        "maxWidthInPixel": 4096,
        "maxHeightInPixel": 3072,
        "sizingPolicy": "keep",
        "playbackSpeed": 1.5
    },
    "watermarkId": "wmk-fekn2ydgq2fe9f1x"
}
```

响应（Response）

- 响应头域：无特殊Header参数
- 响应参数：无
- 响应示例：

```
HTTP/1.1 200 OK
Transfer-Encoding: chunked
x-bce-request-id: 0969b91e-1890-470b-b93b-97939f672e54
Cache-Control: no-cache
Server: BWS
Date: Fri, 14 Feb 2020 07:12:37 GMT
Content-Type: application/json;charset=UTF-8
```

视频转码任务接口

创建视频转码任务

接口描述

用户通过该接口创建转码任务，支持多视频合并。

请求（Request）

- 请求语法：

```
POST /v{version}/job/transcoding HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
connection: keep-alive
accept: */*
host: media.bj.baidubce.com
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

- 请求头域：无特殊Header参数
- 请求参数：

字段名称	字段类型	必要性	字段描述	可选值	默认值
pipelineName	String	必选	任务所属的pipelineName	-	-
source	Object	必选	输入的原始信息的集合	-	-
+ sourceKey	String	可选	原始文件的BOS Key，即是相对于输入Bucket的文件的相对路径	-	-
+ clips	Array	可选	待合并的原始视频信息，与上面的sourceKey不可同时指定，数组长度[1,200]，用法及限制参考 合并多个视频	-	-
++ bucket	String	可选	原始文件的BOS Bucket（用户必须有该bucket的读权限）	-	队列中指定的输入bucket
++ sourceKey	String	必选	原始文件的BOS Key，相对于输入Bucket的文件的相对路径，支持输入为public bucket下的hls（即sourceKey为m3u8文件）	-	-
++ asMasterClip	Bool	可选	是否指定该片段作为主分片，即转码分辨率参考该片段。当clips中存在 asMasterClip为true时，对应模板Preset的 video.sizingPolicy必须为shrinkToFit，此时输出分辨率保持主分片视频宽高比，其他分片参考该分辨率加黑边对齐。	true, false	false
++ enableLogo	Bool	可选	是否允许在该片段添加水印。为空时，如有指定水印，默认该片段添加水印	true, false	true
++ enableDelogo	Bool	可选	是否允许在该片段进行去水印，去水印位置参数为job.target.delogo	true, false	true
++ enableCrop	Bool	可选	是否允许在该片段添进行去黑边，去黑边位置参数为job.target.crop	true, false	true
++ startTimeInSecond	Number	可选	视频片段的起始时间	大于等于0	NULL（从0s开始）

ond					
++ durationInSecond	Number	可选	视频片段的持续时间	大于等于1	NULL (代表从指定开始时间点到视频结结尾的时长)
++ startTimeInMillisecond	Number	可选	视频片段的起始时间，单位毫秒，与startTimeInSecond同时指定时优先生效	大于等于0	NULL（代表从0s开始）
++ durationInMillisecond	Number	可选	视频片段的持续时间，单位毫秒，与durationInSecond同时指定时优先生效	大于等于1	NULL (代表从指定开始时间点到视频结结尾的时长)
target	Object	必选	输出信息的集合	-	-
+ targetBucket	String	可选	目标文件的输出BOS bucket	-	队列中指定的输出bucket
+ targetKey	String	必选	目标文件的BOS key，即是相对于输出Bucket的文件的相对路径	-	-
+ presetName	String	必选	输出处理的模板的presetName	-	-
+ jobCfg	Object	选填	转码任务配置。		
++ notification	String	选填	通知地址，http地址，且不可为localhost。		
+ autoDelogo	Boolean	选填	自动去水印。		
+ delogoMode	String	选填	自动去水印模式	Normal, Inpainting	Normal
+ delogoArea	Object	可选	手动去水印位置设置参数，描述水印位置区域。注：使用delogo去水印功能时，不能指定transmux模式的模板；同时设置autoDelogo时，默认优先自动去水印模式。	-	-
++ x	Number	必选	水印区域相对左上角的x（横）坐标，左上角为0	大于等于0	-
++ y	Number	必选	水印区域相对左上角的y（纵）坐标，左上角为0	大于等于0	-
++ width	Number	必选	水印区域横向宽度	大于等于1	-
++ height	Number	必选	水印区域纵向高度	大于等于1	-
+			手动去水印多个水印位置区域设置参数，至多指定5个。使用		

delogoAreas	Array	可选	delogo去水印功能时，不能指定transmux模式的模板，不可与delogoArea同时指定	-	-
++ x	Number	必选	水印区域相对左上角的x（横）坐标，左上角为0	大于等于0	-
++ y	Number	必选	水印区域相对左上角的y（纵）坐标，左上角为0	大于等于0	-
++ width	Number	必选	水印区域横向宽度	大于等于1	-
++ height	Number	必选	水印区域纵向高度	大于等于1	-
+ autoCrop	Bool	可选	开启自动剪裁黑边。和crop同时设置时，以crop为准	true、false	false
+ crop	Object	可选	黑边裁剪参数，描述除去黑边后的有效区域。使用crop去黑边功能时，不能指定transmux模式的模板	-	-
++ x	Number	必选	去黑边后的有效区域相对左上角的x（横）坐标，左上角为0	大于等于0	-
++ y	Number	必选	去黑边后的有效区域相对左上角的y（纵）坐标，左上角为0	大于等于0	-
++ width	Number	必选	去黑边后的有效区域横向宽度	大于等于1	-
++ height	Number	必选	去黑边后的有效区域纵向高度	大于等于1	-
+ watermarkIds	Array	可选	水印模板id集合，最大size 5。当Job和Preset中同时指定watermarkId(s)时，优先使用Job中设置的watermarkId，以支持更灵活地设置水印	-	-
+ inserts	Array	选填	待插入（叠加）的内容，类型可以为图片、视频、音频、字幕、文本水印等。audio类型的inserts不能和其他类型inserts共存。不支持同时设置水印和inserts。多clips的任务不可设置inserts。数组最大长度为200	-	-
++ bucket	String	选填	BOS存储上insert文件Bucket，type 为 text 时不可设置，否则必须设置	-	-
++ key	String	选填	BOS存储上insert文件Key，type 为 text 时不可设置，否则必须设置	-	-
++ type	String	必选	insert类型，可选值 video, image, audio, subtitle, text，分别表示视频、图片、音频、字幕、文本水印。其中，video类型支持输入key为MOV等格式，image类型支持输入key为JPG、PNG、APNG等格式，audio类型支持输入key为MP3、AAC等格式，subtitle类型支持输入key为srt等格式	video, image, audio, subtitle, text	-
++ text	String	可选	文本水印内容（当且仅当 type 为 text 时设置）	-	-
++ font	Object	可选	字体效果（当且仅当 type 为 subtitle、 text 时设置）	-	-
+++ family	String	可选	字体系列	STXihei	STXihei
+++ sizeInPoint	Number	可选	字体大小	0 ~ 72	0 使用默认大小，即16
+++ color	String	选填	字体颜色		默认颜色为#FFFFFF
++					

layout	Object	可选	显示位置（type为audio时不可设置layout）	-	-
+++ verticalAlignment	String	可选	垂直对齐方式	top, center, bottom	top
+++ horizontalAlignment	String	可选	水平对齐方式	left, center, right	left
+++ verticalOffsetInPixel	Number	可选	垂直偏移，该参数仅在verticalAlignment设置为top或bottom时有效，单位：像素	0~3072	0
+++ horizontalOffsetInPixel	Number	可选	水平偏移，该参数仅在horizontalAlignment设置为left或right时有效，单位：像素	0~4096	0
++ timeline	Object	可选	有效显示起止时间	-	-
+++ startTimeInMillisecond	Number	可选	水印、图片、文本水印等的显示起始时间，单位：豪秒	大于等于0	-
+++ durationInMillisecond	Number	可选	水印、图片、文本水印等的显示持续时间，单位：豪秒	大于等于0	-
+digitalWmTextContent	String	可选	需嵌入的数字水印文字内容	字符数<=100, 支持中英文及标点符号(如需嵌入中文请选择v1及以上版本算法)	-
+digitalWmImageBucket	String	可选	(仅算法0支持图片)需嵌入的数字水印图片的Bos Bucket	-	-
+digitalWmImageKey	String	可选	(仅算法0支持图片)需嵌入的数字水印图片的Bos Key	-	-
+digitalWmSecretKeyId	String	可选	数字水印密钥模板ID，密钥用于对水印加密嵌入，提取水印需提供正确密钥	需从用户已创建的数字水印密钥模板选择，生效优先级高于转码模板中的密钥	-
+digitalWmAlgVersion	Number	可选	算法版本号	0 ~ 2	0

+digitalWmStrength	Number	可选	(算法1、2有效)数字水印嵌入强度，对同一算法，强度越高则抗攻击能力越强，隐蔽性越差	0 ~ 1	0.5

- 请求示例：

```
 POST /v3/job/transcoding HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2015-03-24T13:06:02Z
connection: keep-alive
accept: */*
host: media.bj.baidubce.com
x-bce-request-id: 6d0b0a36-2ffe-49d4-9d81-333a9ab9417e
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2015-03-24T13:06:02Z/1800/host;x-bce-
date/02f64774999996903cffa5ae4d6eef436127a96f581a4e8467497e239d824be8
{
    "pipelineName": "high_priority_pipe",
    "source": {
        "clips": [
            {
                "sourceKey": "input_media1.mp4",
                "startTimeInSecond": 0,
                "durationInSecond": 300
            },
            {
                "sourceKey": "input_media2.mp4"
            },
            {
                "sourceKey": "input_media3.mp4"
            }
        ]
    },
    "target": {
        "targetKey": "SampleOutput.mp4",
        "presetName": "bce.video_mp4_1280x720_1728kbps",
        "delogoArea": {
            "x": 10,
            "y": 20,
            "width": 200,
            "height": 150
        },
        "inserts": [
            {
                "bucket" : "samplebucket",
                "key" : "samplefolderpath/samplepicture.png",
                "type": "image",
                "layout": {
                    "verticalAlignment" : "bottom",
                    "horizontalAlignment" : "left",
                    "verticalOffsetInPixel" : 0,
                    "horizontalOffsetInPixel" : 0,
                },
                "timeline" : {
                    "startTimeInMillisecond" : 5000,
                    "durationInMillisecond" : 65500
                }
            }
        ]
    }
}
```

响应（Response）

- 响应头域：无特殊Header参数
- 响应参数：

字段名称	字段类型	字段描述
jobId	String	系统生成的Job的唯一标示jobId

- 响应示例：

```
 HTTP/1.1 200 OK
Transfer-Encoding: chunked
x-bce-request-id: 6bae5cb3-97d1-4b1a-b8b6-0ad577c1d481
Cache-Control: no-cache
Server: BWS
Date: Tue, 24 Mar 2015 13:34:07 GMT
Content-Type: application/json;charset=UTF-8

{
    "jobId":"job-fczspxdutvmnbamq"
}
```

查询指定队列的视频转码任务信息

接口描述

查询指定队列下满足一定条件的所有转码任务。

请求（Request）

- 请求语法：

```
 GET /v{version}/job/transcoding?pipelineName={pipelineName}&jobStatus={jobStatus}&begin={begin}&end=
{end}}&marker={marker}&maxSize={maxSize} HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

- 请求头域：无特殊Header参数

- 请求参数：

字段名称	字段类型	必要性	字段描述	可选值	默认值
pipelineName	String	必选	任务所属的队列名	-	-
jobStatus	String	可选	所选任务的状态	SUCCESS, FAILED, PENDING, RUNNING	-
begin	String	可选	任务创建时间的上限。所选任务开始时间要大于等于begin	-	-
end	String	可选	任务创建时间的下限，所选任务开始时间要小于等于end	-	-
marker	String	可选	本次请求的marker，标记查询的起始位置，此处为jobId	-	-
maxSize	Number	可选	本次请求返回的任务列表的最大元素个数	1 ~ 1000	1000

- 请求示例：

```
GET /v3/job/transcoding?pipelineName=high_priority_pipe&jobStatus=SUCCESS&begin=2015-06-15T08%3A53%3A42Z&end=2015-06-18T08%3A53%3A42Z&marker=job-feumm9etdd5c9gqv&maxSize=2 HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2015-03-24T13:37:10Z
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: 3807ce30-5264-45f2-9b52-26b78e24a750
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2015-03-24T13:37:10Z/1800/host;x-bce-date/3e1bf9f50ae1fca2d704d61567810dde946fff3ca2e455676455a6f5c8cce596
```

响应（Response）

- 响应头域：无特殊Header参数
- 响应参数：与[创建视频转码任务/请求/请求参数]保持一致，不返回inserts、crop信息，增加以下字段

字段名称	字段类型	字段描述
jobId	String	任务的唯一标示
jobStatus	String	任务状态
error	Object	job失败时的错误信息，jobStatus == Failed时存在
+ code	String	错误码
+ message	String	错误原因
createTime	String	任务创建的时间
startTime	String	任务开始处理的时间
endTime	String	任务完成处理的时间
marker	String	本次请求的marker，标记查询的起始位置，此处为jobId
isTruncated	Bool	指明返回数据是否被截断。true表示本页后面还有数据，即数据未全部返回；false表示已是最后一页，即数据已全部返回
nextMarker	String	获取下一页所需要传递的marker值（此处为jobId），仅当isTruncated为true时（数据未全部返回）出现

- 响应示例：

```
HTTP/1.1 200 OK
Transfer-Encoding: chunked
x-bce-request-id: 6bae5cb3-97d1-4b1a-b8b6-0ad577c1d481
Cache-Control: no-cache
Server: BWS
Date: Tue, 24 Mar 2015 13:34:07 GMT
Content-Type: application/json;charset=UTF-8

{
  "jobs" : [
  {
    "jobId" : "job-feumm9etdd5c9gqv",
    "pipelineName" : "createjob_20383",
    "source" : {
      "sourceKey" : "jobtest.mp3"
    },
    "target" : {
      "targetKey" : "jobtest_result.mp3",
```

```
      "presetName" : "bce.audio_mp3_320kbps"
    },
    "jobStatus" : "SUCCESS",
    "createTime" : "2015-05-19T11:15:58Z",
    "startTime" : "2015-05-19T11:16:02Z",
    "endTime" : "2015-05-19T11:16:04Z"
  },
  {
    "jobId" : "job-feumnkz275ve7eqn",
    "pipelineName" : "createjob_319293",
    "source" : {
      "clips": [
        {
          "sourceKey": "input_media1.mp4",
          "startTimeInSecond": 0,
          "durationInSecond": 300
        },
        {
          "sourceKey": "input_media2.mp4"
        },
        {
          "sourceKey": "input_media3.mp4"
        }
      ]
    },
    "target" : {
      "targetKey" : "jobtest_result.mp4",
      "presetName" : "bce.video_mp4_1280x720_1728kbps",
      "delogoArea": {
        "x": 10,
        "y": 20,
        "width": 200,
        "height": 150
      },
      "watermarkIds":["wmk-hkit7ufxyx733ctw"]
    },
    "jobStatus" : "FAILED",
    "createTime" : "2015-05-19T11:40:39Z",
    "startTime" : "2015-05-19T11:40:43Z",
    "endTime" : "2015-05-19T11:40:45Z",
    "error" : {
      "code" : "ParameterError",
      "message" : "Parameter error"
    }
  }
  ],
  "marker": "job-feumm9etdd5c9gqv",
  "isTruncated": true,
  "nextMarker": "job-gfpj59idrpygsjtw"
}
```


查询指定视频转码任务

接口描述

通过指定jobId查询该任务的信息。

请求（Request）

- 请求语法：

```
GET /v{version}/job/transcoding/{jobId} HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

- 请求头域：无特殊Header参数
- 请求参数：无
- 请求示例：

```
GET /v3/job/transcoding/job-fczspxdutvmnbamq HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2015-03-24T13:06:02Z
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: 3807ce30-5264-45f2-9b52-26b78e24a750
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2015-03-24T13:06:02Z/1800/host;x-bce-date/02f64774999996903cffa5ae4d6eef436127a96f581a4e8467497e239d824be8
```

响应（Response）

- 响应头域：无特殊Header参数
- 响应参数：与[创建视频转码任务/请求/请求参数]保持一致，不返回inserts、crop信息，增加以下字段

字段名称	字段类型	字段描述
jobId	String	任务的唯一标示
jobStatus	String	任务状态
error	Object	job失败时的错误信息，jobStatus == Failed时存在
+ code	String	错误码
+ message	String	错误原因
createTime	String	任务创建的时间
startTime	String	任务开始处理的时间
endTime	String	任务完成处理的时间

- 响应示例：

```
HTTP/1.1 200 OK
Transfer-Encoding: chunked
x-bce-request-id: 6bae5cb3-97d1-4b1a-b8b6-0ad577c1d481
Cache-Control: no-cache
Server: BWS
Date: Tue, 24 Mar 2015 13:34:07 GMT
Content-Type: application/json;charset=UTF-8

{
    "jobId" : "job-feuncr26j02713wc",
    "pipelineName" : "createjob_388070",
    "source" : {
        "sourceKey" : "media/info/jobtest.mp3"
    },
    "target" : {
        "targetKey" : "jobtest_result.mp3",
        "presetName" : "createjob_388070",
        "delogoArea": {
            "x": 10,
            "y": 20,
            "width": 200,
            "height": 150
        },
        "watermarkIds":["wmk-hkit7ufxyx733ctw"]
    },
    "jobStatus" : "FAILED",
    "createTime" : "2015-05-19T12:28:39Z",
    "startTime" : "2015-05-19T12:28:41Z",
    "endTime" : "2015-05-19T12:28:44Z",
    "error" : {
        "code" : "ParameterError",
        "message" : "Parameter error"
    }
}
```

合并多个视频

创建转码任务 API 中增加 clips 数组用于支持视频合并，该功能按照 clips 中的先后顺序合并成一个完整视频。

> **注意：**
>
> - clips和sourceKey不可共存，通过clips可指定1到200个输入片段。
> - 对模板中三种不同尺寸伸缩策略（Keep，ShrinkToFit / ShrinkToFitBlur，Stretch）的处理和限制：
>
> 1. 如果Keep模式，即保持原始的宽高比例
>
> - 输入：此时合并的多个视频必须具有同样的宽高比例，否则报错。
> - 输出：比较所有输入源视频和模板的宽高，从中选择最小的一组宽高作为输出的宽高。
>
> 2. 如果是 ShrinkToFit / ShrinkToFitBlur 模式
>
> - 输入：无限制
> - 输出：Preset指定的宽高，并保持视频源的宽高比，填充黑边。当指定分片asMasterClip参数（该分片为主分片）时，输出分辨率保持主分片宽高比，其他分片参考输出分辨率填充黑边进行对齐。

3. 如果是Stretch模式

- 输入：无限制
- 输出：Preset指定的宽和高，视频拉伸或者收缩。

- 目前仅支持普通队列合并多个视频，加速队列暂不支持。

keep伸缩模式时的处理策略

当选择keep伸缩模式时，视频的宽高比保持不变，对视频进行相应的缩放。缩放分以下两种情况：

1. 输入视频宽高分别小于模板Preset设置的宽高。
    - 处理方式：目标视频保持原始宽高，不进行缩放。
2. 输入视频宽大于模板的宽或输入视频的高大于模板高。
    - 处理方式：选择缩放比例较小的边进行缩放，使得视频的宽或高等于模板的宽或高。

常见错误码和错误信息

转码任务常见的错误码及错误信息如下表所示：

错误码	错误信息	错误信息(控制台显示)
PartialOK	Partially succeeded.	部分成功
ParameterError	The input parameter is invalid. Please check it.	输入参数错误，请检查输入参数
InternalError	Internal error happened. Please retry or raise one ticket on http://ticket.bce.baidu.com/ if it fails again.	内部错误，请重试；若重试失败，请至"http://ticket.bce.baidu.com/" 提交工单
SystemCancel	System cancelled. Please retry.	系统撤销，请重试
InputNotSupported	The input format is not supported.	输入的数据格式不支持
OutputNotSupported	The output format is not supported.	输出的数据格式不支持
InvalidInputData	The input data is invalid. Please check your input data.	无效的输入数据，请检查您的输入视频是否是有效
IncompatibleInputData	Error when handling the media data.	暂不支持的输入数据，请检查您的输入视频是否是有效
VideoEncryptError	Cannot encrypt video.	视频加密失败
WatermarkSettingError	Watermark extend beyond video border.	水印设置错误，水印图片过大或者位置越界
IllegalWatermarkUsage	Cannot overlay watermark into audio only file.	视频无图像，不允许添加水印
InvalidWatermark	Invlid watermark picture.	水印错误，水印图片格式不正确；若仍失败，请至提交工单
NotImplemented	This feature is not implemented yet. You can raise your requirement on http://ticket.bce.baidu.com/	尚未实现，您可以提交工单
TimeOut	Task time out. Please retry or raise one ticket on http://ticket.bce.baidu.com/ if it fails again.	任务超时，请重试；若重试失败，请提交工单
Unknown	Unknown error. Please retry or raise one ticket on http://ticket.bce.baidu.com/ if it fails again.	未知错误，请重试；若重试失败，请提交工单

抽帧模板接口

创建缩略图模板

接口描述

用户可以通过此接口创建缩略图Preset。

请求（Request）

- 请求语法：

```
POST /v{version}/preset/thumbnail HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
connection: keep-alive
accept: */*
host: media.bj.baidubce.com
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

- 请求头域：无特殊Header参数
- 请求参数（以下均为Requestbody参数）：

字段名称	字段类型	必要性	字段描述	可选值	默认值
presetName	String	必选	缩略图模板名称	-	-
description	String	可选	缩略图模板描述	-	-
target	Object	可选	目标缩略图信息的集合	-	-
+ format	String	可选	目标缩略图的格式	jpg、png、mp4、gif、webp	jpg
+ sizingPolicy	String	可选	目标缩略图的尺寸伸缩策略	keep、shrinkToFit、stretch，keep表示保持原始视频宽高比，shrinkToFit表示保持原始视频宽高比并加黑边，stretch表示拉伸原始视频	keep
+ widthInPixel	Number	可选	目标缩略图的宽，如果视频实际分辨率低于目标分辨率则按照实际分辨率输出	10 ~ 2000	600
+ heightInPixel	Number	可选	目标缩略图的高，如果视频实际分辨率低于目标分辨率则按照实际分辨率输出	10 ~ 2000	450
+ frameRate	Number	可选	动图的帧率，仅当format为mp4、gif、webp且mode为manual、split时可选	0.01 ~ 30.0	3.0
+ gifQuality	String	可选	gif的质量，仅当format为gif且mode为	high, medium	medium

			manual、split时可选		
+ spriteOutputCfg	Object	可选	雪碧图输出参数设置，仅当抽取多图（即mode=manual/split）,且输出为非动图（即format=jpg/png）时可选	-	-
++ rows	Number	可选	雪碧图拼接行数	1 ~ 100	10
++ columns	Number	可选	雪碧图拼接列数	1 ~ 100	10
++ margin	Number	可选	外框宽度，单位：px	1 ~ 1000	0
++ padding	Number	可选	外框宽度，单位：px	1 ~ 1000	0
++ keepCellPic	Bool	可选	是否保留子图原图	true, false	true
++ spriteKeyTag	String	可选	上传BOS的雪碧图的key中用于标记为雪碧图的tag，最终文件名为{keyPrefix}+{spriteKeyTag}+{雪碧图序号%05d}，雪碧图中子图按照原视频中的顺序排列	字符串长度范围为1 ~ 100	"-SPRITE-"
capture	Object	可选	生成缩略图的规则	-	-
+ mode	String	可选	生成缩略图的模式	auto、manual、split、shot、idl、highlight，auto模式是系统自动截取熵值较高的一帧作为缩略图，manual模式是根据指定的起止时间和间隔时间截取缩略图，split模式是根据指定的起止时间和张数截取缩略图，shot模式根据场景切换自动截取画面(不支持输出视频格式)，idl模式时使用百度IDL（Institute of Deep Learning）智能缩略图算法截取缩略图（仅支持输出jpg格式），highlight模式自动生成一个0.5s的精彩片段（目前仅适用于竖屏小	auto

				视频，只支持输出视频格式，默认为正播反播合并效果）	
+ frameNumber	Number	可选	生成缩略图的张数，仅当mode=split时可选	大于等于1	1
+ startTimeInSecond	Number	可选	生成缩略图的开始时间，当mode=manual或split时可选	大于等于0	0.0
+ endTimeInSecond	Number	可选	生成缩略图的结束时间，当mode=manual或split时可选，且不得小于start time	大于等于0	视频时长
+ intervalInSecond	Number	可选	生成缩略图的间隔时间，仅当mode=manual时可选	大于0	1.0
+ minIntervalInSecond	Number	可选	生成缩略图的最小间隔时间，仅当mode=split时可选	大于0	1.0
+ skipBlackFrame	Boolean	可选	是否跳过黑帧，仅当mode=manual或split时可选	true/false	false
+ highlightOutputCfg	Object	可选	highlight 模式下输出控制参数，仅当mode=highlight时可选	-	-
++ durationInSecond	Number	可选	截取片段时长，单位：秒	0.1 ~ 7200.0	0.25
++ playbackSpeed	Number	可选	回放速度，值低于1.0时为减速视频，高于1.0	0.05 ~ 20.0	1.0

			频，高于1.0时为加速视频		
++ frameRate	Number	可选	输出视频帧率，单位：fps	0.1 ~ 60.0	11.0
++ reverseConcat	Bool	可选	正播反播合并效果	true, false	true

- 请求示例：

```
POST /v3/preset/thumbnail HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2015-03-24T13:08:44Z
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: 3807ce30-5264-45f2-9b52-26b78e24a750
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2015-03-24T13:08:44Z/1800/host;x-bce-date/7e21c9cf1e4e2cc6921a407a388fe98df122c53b9f509043d841be76eb09a1f9

{
    "presetName": "customlized_thumbnail_preset",
    "description": "An example thumbnail preset description",
    "container": "mp4",
    "state": "ACTIVE",
    "target": {
        "format": "jpg",
        "sizingPolicy": "keep",
        "widthInPixel": 600,
        "heightInPixel": 450
    },
    "capture": {
        "mode": "manual",
        "startTimeInSecond": 0,
        "endTimeInSecond": 1
    }
}
```

响应（Response）

- 响应头域：无特殊Header参数
- 响应参数：无
- 响应示例：

```
HTTP/1.1 200 OK
Transfer-Encoding: chunked
x-bce-request-id: 3807ce30-5264-45f2-9b52-26b78e24a750
Cache-Control: no-cache
Server: BWS
Date: Tue, 24 Mar 2015 13:37:10 GMT
Content-Type: application/json;charset=UTF-8
```

查询指定模板

接口描述

通过presetName查询指定的缩略图模板信息。

请求（Request）

- 请求语法：

```
GET /v{version}/preset/thumbnail/{presetId} HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

- 请求头域：无特殊Header参数
- 请求参数：无
- 请求示例：

```
GET /v3/preset/thumbnail/customlized_thumbnail_preset HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2015-03-24T13:37:10Z
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: 3807ce30-5264-45f2-9b52-26b78e24a750
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2015-03-24T13:37:10Z/1800/host;x-bce-date/3e1bf9f50ae1fca2d704d61567810dde946fff3ca2e455676455a6f5c8cce596
```

响应（Response）

- 响应头域：无特殊Header参数
- 响应参数：与[创建缩略图模板/请求/请求参数]保持一致，增加以下字段

字段名称	字段类型	字段描述
state	String	模板状态，ACTIVE/INACTIVE
createdTime	String	模板创建的UTC格式的时间

- 响应示例：

```
HTTP/1.1 200 OK
Transfer-Encoding: chunked
x-bce-request-id: 3807ce30-5264-45f2-9b52-26b78e24a750
Cache-Control: no-cache
Server: BWS
Date: Tue, 24 Mar 2015 13:37:10 GMT
Content-Type: application/json;charset=UTF-8

{
    "state": "ACTIVE",
    "createdTime": "2015-03-24T13:34:07Z",
    "presetName": "customlized_thumbnail_preset",
    "description": "A example thumbnail preset description",
    "target": {
    "format": "jpg",
    "sizingPolicy": "keep",
    "widthInPixel": 600,
    "heightInPixel": 450
    },
    "capture": {
        "mode": "manual",
        "startTimeInSecond": 0,
        "endTimeInSecond": 1
    }
}
```

删除指定缩略图模板

接口描述

用于删除用户指定presetName的用户缩略图模板

请求（Request）

- 请求语法：

```
DELETE /v{version}/preset/thumbnail/{presetName} HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
connection: keep-alive
accept: */*
host: media.bj.baidubce.com
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

- 请求头域：无特殊Header参数
- 请求参数：无
- 请求示例：

```
DELETE /v3/preset/thumbnail/customlized_thumbnail_preset HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2015-03-24T13:06:02Z
connection: keep-alive
accept: */*
host: media.bj.baidubce.com
x-bce-request-id: 6d0b0a36-2ffe-49d4-9d81-333a9ab9417e
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2015-03-24T13:06:02Z/1800/host;x-bce-date/02f64774999996903cffa5ae4d6eef436127a96f581a4e8467497e239d824be8
```

响应（Response）

- 响应头域：无特殊Header参数
- 响应参数：无
- 响应示例：

```
HTTP/1.1 200 OK
x-bce-request-id: 6d0b0a36-2ffe-49d4-9d81-333a9ab9417e
Cache-Control: no-cache
```

更新指定模板

接口描述

用户可以通过此接口更新指定Preset。

请求（Request）

- 请求语法：

```
PUT /v{version}/preset/thumbnail/{presetName} HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
connection: keep-alive
accept: */*
host: media.bj.baidubce.com
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

- 请求头域：无特殊Header参数
- 请求参数：同创建模板请求
- 请求示例：

```
PUT /v3/preset/thumbnail/customlized_thumbnail_preset HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2020-02-14T15:12:37Z
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: 0969b91e-1890-470b-b93b-97939f672e54
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2020-02-14T15:12:37Z/1800/host;x-bce-date/7e21c9cf1e4e2cc6921a407a388fe98df122c53b9f509043d841be76eb09a1f9

{
    "presetName": "customlized_thumbnail_preset",
    "description": "An example thumbnail preset description",
    "target": {
    "format": "jpg",
    "sizingPolicy": "keep",
    "widthInPixel": 600,
    "heightInPixel": 450
    },
    "capture": {
        "mode": "manual",
        "startTimeInSecond": 0,
        "endTimeInSecond": 1
    }
}
```

响应（Response）

- 响应头域：无特殊Header参数
- 响应参数：无
- 响应示例：

```
HTTP/1.1 200 OK
Transfer-Encoding: chunked
x-bce-request-id: 0969b91e-1890-470b-b93b-97939f672e54
Cache-Control: no-cache
Server: BWS
Date: Fri, 14 Feb 2020 07:12:37 GMT
Content-Type: application/json;charset=UTF-8
```

抽帧任务接口

创建缩略图任务

接口描述

用户通过Bucket，BOS Key以及其他配置信息为指定媒体生成缩略图。

请求（Request）

- 请求语法：

```
POST /v{version}/job/thumbnail HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
connection: keep-alive
accept: */*
host: media.bj.baidubce.com
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

- 请求头域：无特殊Header参数
- 请求参数（均为requestbody参数）：

字段名称	字段类型	必要性	字段描述	可选值	默认值
pipelineName	String	必选	任务所属的pipelineName	-	-
presetName	String	可选	任务的模板名称（模板和job中重复内容以job中为准）	-	-
source	Object	必选	输入视频信息的集合	-	-
+ sourceBucket	String	可选	输入视频文件的BOS Bucket（用户必须有该bucket的读权限）	-	队列中指定的输入bucket
+ key	String	必选	输入视频文件的BOS Key，支持public bucket下的hls（即key为m3u8文件）	-	-
target	Object	可选	目标缩略图信息的集合	-	-
+ targetBucket	String	可选	目标缩略图的BOS Bucket	-	队列中指定的输出bucket
+ keyPrefix	String	可选	目标缩略图的BOS Key的前缀，mode=auto/idl/highlight 时目标文件名为{keyPrefix}，mode=manual/split/shot时目标文件名为{keyPrefix}加截图时间点	-	{sourceKey}
+ format	String	可选	目标缩略图的格式	jpg、png、mp4、gif、webp	jpg

+ sizingPolicy	String	可选	目标缩略图的尺寸伸缩策略	keep、shrinkToFit、stretch，keep表示保持原始视频宽高比，shrinkToFit表示保持原始视频宽高比并加黑边，stretch表示拉伸原始视频	keep
+ widthInPixel	Number	可选	目标缩略图的宽，如果视频实际分辨率低于目标分辨率则按照实际分辨率输出	10 ~ 2000	600
+ heightInPixel	Number	可选	目标缩略图的高，如果视频实际分辨率低于目标分辨率则按照实际分辨率输出	10 ~ 2000	450
+ frameRate	Number	可选	动图的帧率，仅当format为mp4、gif、webp且mode为manual、split时可选	0.01 ~ 30.0	3.0
+ gifQuality	String	可选	gif的质量，仅当format为gif且mode为manual、split时可选	high, medium	medium
+ spriteOutputCfg	Object	可选	雪碧图输出参数设置，仅当抽取多图（即mode=manual/split），且输出为非动图（即format=jpg/png）时可选	-	-
++ rows	Number	可选	雪碧图拼接行数	1 ~ 100	10
++ columns	Number	可选	雪碧图拼接列数	1 ~ 100	10
++ margin	Number	可选	外框宽度，单位：px	1 ~ 1000	0
++ padding	Number	可选	外框宽度，单位：px	1 ~ 1000	0
++ keepCellPic	Bool	可选	是否保留子图原图	true, false	true
++ spriteK	String	可选	上传BOS的雪碧图的key中用于标记为雪碧图的tag，最终文件名为{keyPrefix}+	字符串长度范围为1 ~ 100	"-SPRITE-

eyTag			{spriteKeyTag}+{雪碧图序号%05d}，雪碧图中子图按照原视频中的顺序排列		"
capture	Object	可选	生成缩略图的规则	-	-
+ mode	String	可选	生成缩略图的模式	auto、manual、split、shot、idl、highlight，auto模式是系统自动截取熵值较高的一帧作为缩略图，manual模式是根据指定的起止时间和间隔时间截取缩略图，split模式是根据指定的起止时间和张数截取缩略图，shot模式根据场景切换自动截取画面(不支持输出视频格式)，idl模式时使用百度IDL（Institute of Deep Learning）智能缩略图算法截取缩略图（仅支持输出jpg、png格式），highlight模式自动生成一个0.5s的精彩片段（目前仅适用于竖屏小视频，只支持输出视频格式，默认为正播反播合并效果）	auto
+ skipBlackFrame	Boolean	可选	是否跳过黑帧，仅当mode=manual或split时可选	true/false	false
+ frameNumber	Number	可选	生成缩略图的张数，仅当mode=split时可选	大于等于1	1
+ startTimeInSecond	Number	可选	生成缩略图的开始时间，当mode=manual或split时可选	大于等于0	0.0
+ endTimeInSecond	Number	可选	生成缩略图的结束时间，当mode=manual或split时可选，且不得小于start time	大于等于0	视频时长
+ intervallnSecond	Number	可选	生成缩略图的间隔时间，仅当mode=manual时可选	大于0	1.0
+ minIntervalInSecond	Number	可选	生成缩略图的最小间隔时间，仅当mode=split时可选	大于0	1.0
+ highlightOutputCfg	Object	可选	highlight 模式下输出控制参数，仅当mode=highlight时可选	-	-
++ durationInSecond	Number	可选	截取片段时长，单位：秒	0.1 ~ 7200.0	0.25
++ playbac	Number	可选	回放速度，值低于1.0时为减速视频，	0.05 ~ 20.0	1.0

playbackSpeed	Number	可选	高于1.0时为加速视频		
++ frameRate	Number	可选	输出视频帧率，单位：fps	0.1 ~ 60.0	11.0
++ reverseConcat	Bool	可选	正播反播合并效果	true, false	true
delogoArea	Object	可选	去水印参数，描述水印位置区域。	-	-
+ x	Number	必选	水印区域相对左上角的x（横）坐标，左上角为0	大于等于0	-
+ y	Number	必选	水印区域相对左上角的y（纵）坐标，左上角为0	大于等于0	-
+ width	Number	必选	水印区域横向宽度	大于等于1	-
+ height	Number	必选	水印区域纵向高度	大于等于1	-
delogoAreas	Array	可选	去多个水印参数，描述水印位置区域。不可与delogoArea同时指定	-	-
+ x	Number	必选	水印区域相对左上角的x（横）坐标，左上角为0	大于等于0	-
+ y	Number	必选	水印区域相对左上角的y（纵）坐标，左上角为0	大于等于0	-
+ width	Number	必选	水印区域横向宽度	大于等于1	-
+ height	Number	必选	水印区域纵向高度	大于等于1	-
crop	Object	可选	黑边裁剪参数，描述除去黑边后的有效区域（不可同时设置crop和shrinkToFit）	-	-
+ x	Number	必选	去黑边后的有效区域相对左上角的x（横）坐标，左上角为0	大于等于0	-
+ y	Number	必选	去黑边后的有效区域相对左上角的y（纵）坐标，左上角为0	大于等于0	-

+ width	Number	必选	去黑边后的有效区域横向宽度	大于等于1	-
+ height	Number	必选	去黑边后的有效区域纵向高度	大于等于1	-

- 请求示例：

```
POST /v3/job/thumbnail HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2015-03-24T13:08:44Z
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: d97c57d0-ca44-4d1c-bfeb-941a92440968
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2015-03-24T13:08:44Z/1800/host;x-bce-date/7e21c9cf1e4e2cc6921a407a388fe98df122c53b9f509043d841be76eb09a1f9

{
    "pipelineName": "high_priority_pipe",
    "source": {
        "key": "samplefolderpath/samplevideo.mp4"
    },
    "target": {
        "keyPrefix": "sampleoutputfolderpath/samplethumbnail-",
        "format": "jpg",
        "sizingPolicy": "keep",
        "widthInPixel": "600",
        "heightInPixel": "450"
    },
    "capture": {
        "mode": "manual",
        "startTimeInSecond": "20",
        "endTimeInSecond": "50",
        "intervalInSecond": "10"
    },
    "delogoArea": {
        "x": 10,
        "y": 20,
        "width": 200,
        "height": 150
    }
}
```

响应（Reponse）

- 响应头域：无特殊Header参数
- 响应参数：

字段名称	字段类型	字段描述
jobId	String	系统生成的Thumbnail的唯一标示thumbnailId

- 响应示例：

```
 HTTP/1.1 200 OK
Transfer-Encoding: chunked
x-bce-request-id: d97c57d0-ca44-4d1c-bfeb-941a92440968
Cache-Control: no-cache
Server: BWS
Date: Tue, 24 Mar 2015 13:34:07 GMT
Content-Type: application/json;charset=UTF-8

{
    "jobId":"job-lsdspxdastsmnbam"
}
```

查询指定缩略图任务

请求（Request）

- 请求语法：

```
 GET /v{version}/job/thumbnail/{jobId} HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

- 请求头域：无特殊Header参数
- 请求参数：无
- 请求示例：

```
 GET /v3/job/thumbnail/job-lsdspxdastsmnbam HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2015-03-24T13:08:44Z
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: 6bae5cb3-97d1-4b1a-b8b6-0ad577c1d481
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2015-03-24T13:08:44Z/1800/host;x-bce-
date/7e21c9cf1e4e2cc6921a407a388fe98df122c53b9f509043d841be76eb09a1f9
```

响应（Reponse）

- 响应头域：无特殊Header参数
- 响应参数：与[创建缩略图任务/请求/请求参数]保持一致，增加以下字段

字段名称	字段类型	字段描述
jobId	String	缩略图的唯一标识
jobStatus	String	缩略图生成状态
target		
+ keys	Array	目标缩略图的BOS的Key的集合
createTime	String	任务开始时间
endTime	String	任务完成时间

- 响应示例：

```
 HTTP/1.1 200 OK
Transfer-Encoding: chunked
x-bce-request-id: 6bae5cb3-97d1-4b1a-b8b6-0ad577c1d481
Cache-Control: no-cache
Server: BWS
Date: Tue, 24 Mar 2015 13:34:07 GMT
Content-Type: application/json;charset=UTF-8

{
    "jobId": "job-lsdspxdastsmnbwx",
    "jobStatus": "success",
    "pipelineName": "high_priority_pipe",
    "source": {
        "key": "samplefolderpath/samplevideo.mp4"
    },
    "target": {
        "keyPrefix": "sampleoutputfolderpath/samplethumbnail-",
        "keys": [
            "sampleoutputfolderpath/samplethumbnail-00020000.jpg",
            "sampleoutputfolderpath/samplethumbnail-00030000.jpg",
            "sampleoutputfolderpath/samplethumbnail-00040000.jpg",
            "sampleoutputfolderpath/samplethumbnail-00050000.jpg"
        ],
        "format": "jpg",
        "sizingPolicy": "keep",
        "widthInPixel": "600",
        "heightInPixel": "450"
    },
    "capture": {
        "mode": "manual",
        "startTimeInSecond": "20",
        "endTimeInSecond": "50",
        "intervalInSecond": "10"
    },
    "delogoArea": {
        "x": 10,
        "y": 20,
        "width": 200,
        "height": 150
    }
}
```

查询指定队列的缩略图任务信息

请求（Request）

- 请求语法：

```
GET /v{version}/job/thumbnail?pipelineName={pipelineName}&jobStatus={jobStatus}&begin={begin}&end={end}}&marker={marker}&maxSize={maxSize} HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

- 请求头域：无特殊Header参数
- 请求参数（均为query参数）：

字段名称	字段类型	必要性	字段描述	可选值	默认值
pipelineName	String	必选	任务所属的队列名	-	-
jobStatus	String	可选	所选任务的状态	SUCCESS, FAILED, PENDING, RUNNING	-
begin	String	可选	任务创建时间的上限。所选任务开始时间要大于等于begin	-	-
end	String	可选	任务创建时间的下限，所选任务开始时间要小于等于end	-	-
marker	String	可选	本次请求的marker，标记查询的起始位置，此处为jobId	-	-
maxSize	Number	可选	本次请求返回的任务列表的最大元素个数	1 ~ 1000	1000

- 请求示例：

```
GET /v3/job/thumbnail?pipelineName=high_priority_pipe&jobStatus=SUCCESS&begin=2015-06-15T08%3A53%3A42Z&end=2015-06-19T08%3A53%3A42Z&marker=job-feumm9etdd5c9gqv HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2015-03-24T13:08:44Z
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: d97c57d0-ca44-4d1c-bfeb-941a92440968
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2015-03-24T13:08:44Z/1800/host;x-bce-date/7e21c9cf1e4e2cc6921a407a388fe98df122c53b9f509043d841be76eb09a1f9
```

响应（Reponse）

- 响应头域：无特殊Header参数
- 响应参数：与[创建缩略图任务/请求/请求参数]保持一致，增加以下字段

字段名称	字段类型	字段描述
jobId	String	缩略图的唯一标识
jobStatus	String	缩略图生成状态
target		
+ keys	Array	目标缩略图的BOS的Key的集合
createTime	String	任务开始时间
endTime	String	任务完成时间
marker	String	本次请求的marker，标记查询的起始位置，此处为jobId
isTruncated	Bool	指明返回数据是否被截断。true表示本页后面还有数据，即数据未全部返回；false表示已是最后一页，即数据已全部返回
nextMarker	String	获取下一页所需要传递的marker值（此处为jobId），仅当isTruncated为true时（数据未全部返回）出现

- 响应示例：

```
 HTTP/1.1 200 OK
Transfer-Encoding: chunked
x-bce-request-id: d97c57d0-ca44-4d1c-bfeb-941a92440968
Cache-Control: no-cache
Server: BWS
Date: Tue, 24 Mar 2015 13:34:07 GMT
Content-Type: application/json;charset=UTF-8

{
  "thumbnails": [
  {
    "createTime" : "2015-05-19T11:15:58Z",
    "endTime" : "2015-05-19T11:16:04Z",
    "jobId": "job-fgcivp49gvnggqq8",
    "jobStatus": "success",
    "pipelineName": "high_priority_pipe",
    "source": {
      "key": "samplefolderpath/samplevideo.mp4"
    },
    "target": {
      "keyPrefix": "sampleoutputfolderpath/samplethumbnail-",
      "keys": [
        "sampleoutputfolderpath/samplethumbnail-00045000.jpg"
      ],
      "format": "jpg",
      "sizingPolicy": "keep",
      "widthInPixel": "600",
      "heightInPixel": "450"
    },
    "capture": {
      "mode": "manual",
      "startTimeInSecond": "45",
      "endTimeInSecond": "50",
      "intervalInSecond": "10"
    }
  },
  {
    "createTime" : "2015-05-19T109:15:58Z",
    "endTime" : "2015-05-19T09:35:04Z",
    "jobId": "job-lsdspxdastsmnbwx",
```

```
    "jobStatus": "success",
    "pipelineName": "high_priority_pipe",
    "source": {
      "key": "samplefolderpath/samplevideo.mp4"
    },
    "target": {
      "keyPrefix": "sampleoutputfolderpath/samplethumbnail",
      "keys": [
        "sampleoutputfolderpath/samplethumbnail.jpg"
      ],
      "format": "jpg",
      "sizingPolicy": "shrinkToFit",
      "widthInPixel": "800",
      "heightInPixel": "600"
    },
    "capture": {
       "mode": "auto"
    }
  },
  {
    "createTime" : "2015-05-19T109:14:58Z",
    "endTime" : "2015-05-19T09:35:04Z",
    "jobId": "job-lsdspxiwpxsmnber",
    "jobStatus": "running",
    "pipelineName": "high_priority_pipe",
    "source": {
      "key": "samplefolderpath/samplevideo.mp4"
    },
    "target": {
      "keyPrefix": "samplefolderpath/samplevideo",
      "format": "jpg",
      "sizingPolicy": "keep",
      "widthInPixel": "1280",
      "heightInPixel": "450"
    },
    "capture": {
      "mode": "auto"
    }
  },
  {
    "createTime" : "2015-05-19T109:10:58Z",
    "endTime" : "2015-05-19T09:35:04Z",
    "jobId": "job-lsdspxiwpxsmnbeq",
    "jobStatus": "failed",
    "pipelineName": "high_priority_pipe",
    "source": {
      "key": "samplefolderpath/samplevideo.mp4"
    },
    "target": {
      "keyPrefix": "samplefolderpath/samplethumbnail-",
      "format": "jpg",
      "sizingPolicy": "keep",
      "widthInPixel": "1280",
      "heightInPixel": "450"
    },
    "capture": {
      "mode": "manual",
      "startTimeInSecond": "10"
    },
    "error": {
      "code": "InvalidInputData",
      "message": "The input data is invalid, please ..."
```

```
    }
  }
  ],
  "marker": "job-fgcivp49gvnggqq8",
  "isTruncated": false
}
```

视频质量检测模板接口

创建视频质检模板

接口描述

用户可以通过此接口创建视频质检模板。

请求（Request）

- 请求语法：

```
POST /v{version}/preset/video_defect_detect HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
connection: keep-alive
accept: */*
host: media.bj.baidubce.com
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

- 请求头域：无特殊Header参数
- 请求参数：无
- 请求体：

字段名称	字段类型	必要性	字段描述	可选值	默认值
presetName	String	必选	模板名称	-	-
description	String	可选	模板描述	-	-
models	ModelsParm	可选	需要检测的项目（使用的模型）	-	-

ModelsParm

字段名称	字段类型	必要性	字段描述	可选值	默认值
whiteScreen	ModelParm	可选	白屏检测模型	-	-
blackScreen	ModelParm	可选	黑屏检测模型	-	-
tooBright	ModelParm	可选	过亮检测模型	-	-
tooDark	ModelParm	可选	过暗检测模型	-	-
redScreen	ModelParm	可选	红屏检测模型	-	-
yellowScreen	ModelParm	可选	黄屏检测模型	-	-
greenScreen	ModelParm	可选	绿屏检测模型	-	-
blueScreen	ModelParm	可选	蓝屏检测模型	-	-
purpleScreen	ModelParm	可选	紫屏检测模型	-	-
reddish	ModelParm	可选	偏红检测模型	-	-
yellowish	ModelParm	可选	偏黄检测模型	-	-
greenish	ModelParm	可选	偏绿检测模型	-	-
bluish	ModelParm	可选	偏蓝检测模型	-	-
purplish	ModelParm	可选	偏紫检测模型	-	-
blur	ModelParm	可选	模糊检测模型	-	-
noise	ModelParm	可选	噪声检测模型	-	-
mosaic	ModelParm	可选	马赛克检测模型	-	-
freeze	ModelParm	可选	冻结检测模型	-	-
jitter	ModelParm	可选	抖动检测模型	-	-
blackEdge	ModelParm	可选	黑边检测模型	-	-
blurEdge	ModelParm	可选	模糊边缘检测模型	-	-
staticEdge	ModelParm	可选	静态边缘检测模型	-	-
crash	ModelParm	可选	花屏检测模型	-	-
colorBar	ModelParm	可选	彩条检测模型	-	-
block	ModelParm	可选	块效应检测模型	-	-
interlace	ModelParm	可选	场效应检测模型	-	-
mute	ModelParm	可选	静音检测模型	-	-
volumeLow	ModelParm	可选	音量过低检测模型	-	-
volumeHigh	ModelParm	可选	音量过高检测模型	-	-
soundIntermittent	ModelParm	可选	声音间断检测模型	-	-

ModelParm

字段名称	字段类型	必要性	字段描述	可选值	默认值
enable	Boolean	可选	是否使用该模型	true、false	true
interval	Number	可选	检测间隔（毫秒）	正整数	jitter模型为200，mute/volumeLow/volumeHigh/soundIntermittent模型为100，其他模型均为1000
threshold	Number	可选	检测分值的阈值，对应单帧分值>=threshold时，认为单帧异常。tooBright/tooDark/reddish/yellowish/greenish/bluish/purplish/blur/noise/mosaic/blackEdge/blurEdge/staticEdge/block/volumeLow/volumeHigh/soundIntermittent模型支持阈值自定义，其他模型不支持	0.0~1.0	tooBright/tooDark/blur/noise/reddish/yellowish/greenish/bluish/purplish模型为0.5，mosaic模型为0.001，blackEdge/blurEdge/staticEdge模型为0.2, block模型为0.4, mute/soundIntermittent模型为0.0001, volumeLow模型为0.01, volumeHigh模型为0.6
duration	Number	可选	持续时长（毫秒）的阈值，对应单帧异常持续时长>=duration时，将对应异常时间段写入检测结果中。	非负整数	blur/noise/freeze/volumeLow模型为2000，soundIntermittent模型为100，volumeHigh模型为200，其他模型均为1000

各模型threshold阈值含义

模型	模型描述	默认阈值	阈值含义
whiteScreen	白屏检测模型	-	-
blackScreen	黑屏检测模型	-	-
tooBright	过亮检测模型	0.500	• 表示检测每帧图像过亮程度（值越大程度越大）的阈值，大于阈值认为单帧存在该异常； • 支持阈值自定义[0,1]，阈值越大，表示检测门槛越高。
tooDark	过暗检测模型	0.500	• 表示检测每帧图像过暗程度（值越大程度越大）的阈值，大于阈值认为单帧存在该异常； • 支持阈值自定义，阈值越大，表示检测门槛越高。
redScreen	红屏检测模型	-	-
yellowScreen	黄屏检测模型	-	-
greenScreen	绿屏检测模型	-	-
blueScreen	蓝屏检测模型	-	-
purpleScree	紫屏检测模	-	-

n	型		
reddish	偏红检测模型	0.500	• 表示检测每帧图像偏红程度（值越大程度越大）的阈值，大于阈值认为单帧存在该异常； • 支持阈值自定义[0,1]，设定阈值越大，表示检测门槛越高。
yellowish	偏黄检测模型	0.500	• 表示检测每帧图像偏黄程度（值越大程度越大）的阈值，大于阈值认为单帧存在该异常； • 支持阈值自定义[0,1]，设定阈值越大，表示检测门槛越高。
greenish	偏绿检测模型	0.500	• 表示检测每帧图像偏绿程度（值越大程度越大）的阈值，大于阈值认为单帧存在该异常； • 支持阈值自定义[0,1]，设定阈值越大，表示检测门槛越高。
bluish	偏蓝检测模型	0.500	• 表示检测每帧图像偏蓝程度（值越大程度越大）的阈值，大于阈值认为单帧存在该异常； • 支持阈值自定义[0,1]，设定阈值越大，表示检测门槛越高。
purplish	偏紫检测模型	0.500	• 表示检测每帧图像偏紫程度（值越大程度越大）的阈值，大于阈值认为单帧存在该异常； • 支持阈值自定义[0,1]，设定阈值越大，表示检测门槛越高。
blur	模糊检测模型	0.500	• 表示检测每帧图像模糊程度（值越大程度越大）的阈值，大于阈值认为单帧存在该异常； • 支持阈值自定义[0,1]，设定阈值越大，表示检测门槛越高。
noise	噪声检测模型	0.500	• 表示检测每帧图像噪声程度（值越大程度越大）的阈值，大于阈值认为单帧存在该异常； • 支持阈值自定义[0,1]，设定阈值越大，表示检测门槛越高。
mosaic	马赛克检测模型	0.001	• 表示检测每帧图像中马赛克区域与整个图像面积比值的阈值，大于阈值认为单帧存在该异常； • 支持阈值自定义[0,1]，设定阈值越大，表示检测门槛越高。
freeze	冻结检测模型	-	-
jitter	抖动检测模型	-	-
blackEdge	黑边检测模型	0.200	• 表示检测每帧图像中黑边区域与整个图像面积比值的阈值，大于阈值认为单帧存在该异常； • 支持阈值自定义[0,1]，设定阈值越大，表示检测门槛越高。

blurEdge	模糊边缘检测模型	0.200	• 表示检测每帧图像中模糊边缘区域与整个图像面积比值的阈值，大于阈值认为单帧存在该异常； • 支持阈值自定义[0,1]，设定阈值越大，表示检测门槛越高。
staticEdge	静态边缘检测模型	0.200	• 表示检测每帧图像中静态边缘区域与整个图像面积比值的阈值，大于阈值认为单帧存在该异常； • 支持阈值自定义[0,1]，设定阈值越大，表示检测门槛越高。
crash	花屏检测模型	-	-
colorBar	彩条检测模型	-	-
block	块效应检测模型	0.400	• 表示检测每帧图像块效应程度（值越大程度越大）的阈值，大于阈值认为单帧存在该异常； • 支持阈值自定义[0,1]，设定阈值越大，表示检测门槛越高。
interlace	场效应检测模型	-	-
mute	静音检测模型	0.0001	• 表示检测的每帧音频的音量均值的阈值，小于阈值认为单帧存在该异常； • 支持阈值自定义[0,1]，设定阈值越大，表示检测门槛越高。
volumeLow	音量过低检测模型	0.001	• 表示检测的每帧音频的音量均值的阈值，小于阈值认为单帧存在该异常； • 支持阈值自定义[0,1]，设定阈值越大，表示检测门槛越高。
volumeHigh	音量过高检测模型	0.600	• 表示检测的每帧音频的音量均值的阈值，大于阈值认为单帧存在该异常； • 支持阈值自定义[0,1]，设定阈值越大，表示检测门槛越高。
soundIntermittent	声音间断检测模型	0.0001	• 表示检测的每帧音频的音量均值的阈值，小于阈值认为单帧存在静音，5秒内出现两次及以上静音，则认为存在该异常； • 支持阈值自定义[0,1]，设定阈值越大，表示检测门槛越高。

- 请求示例

```
POST /v3/preset/video_defect_detect HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2021-05-21T21:13:00Z
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: d97c57d0-ca44-4d1c-bfeb-941a92440968
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2021-05-21T21:13:00Z/1800/host;x-bce-date/7e21c9cf1e4e2cc6921a407a388fe98df122c53b9f509043d841be76eb09a1f9

{
    "presetName": "customlized_video_defect_detect_preset",
    "description": "An example video_defect_detect preset description",
    "models": {
        "tooBright": {
            "enable": true,
            "interval": 1000,
            "threshold": 0.5,
            "duration": 2000
        },
        "reddish": {
            "enable": true,
            "interval": 1000,
            "threshold": 0.6,
            "duration": 2000
        },
        "mosaic": {
            "enable": true,
            "interval": 1000,
            "duration": 2000
        }
    }
}
```

响应（Reponse）

- 响应头域：无特殊Header参数
- 响应参数：无
- 响应体：无
- 响应示例：

```
HTTP/1.1 200 OK
Transfer-Encoding: chunked
x-bce-request-id: d97c57d0-ca44-4d1c-bfeb-941a92440968
Cache-Control: no-cache
Server: BWS
Date: Tue, 21 May 2021 21:13:02 GMT
Content-Type: application/json;charset=UTF-8
```

查询指定视频质检模板

接口描述

通过presetName查询视频质检模板。

请求（Request）

- 请求语法：

```
GET /v{version}/preset/video_defect_detect/{presetName} HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

- 请求头域：无特殊Header参数
- 请求参数：

字段名称	字段类型	必要性	字段描述	可选值	默认值
presetName	String	可选	模板名称	-	-

- 请求体：无
- 请求示例：

```
GET /v3/preset/video_defect_detect/customlized_video_defect_detect_preset HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2021-05-21T21:21:21Z
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: 6bae5cb3-97d1-4b1a-b8b6-0ad577c1d481
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2021-05-21T21:21:21Z/1800/host;x-bce-date/7e21c9cf1e4e2cc6921a407a388fe98df122c53b9f509043d841be76eb09a1f9
```

响应（Reponse）

- 响应头域：无特殊Header参数
- 响应参数：无
- 响应体：与[创建视频质检模板/请求/请求体]保持一致，增加以下字段

字段名称	字段类型	字段描述
state	String	模板状态，ACTIVE/INACTIVE，分别表示 在用的模板/已删除的模板
presetType	String	模板类型，SYSTEM/CUSTOM，分别表示 系统内置模板/用户自定义模板
createTime	String	模板创建的UTC格式的时间

- 响应示例：

```
HTTP/1.1 200 OK
Transfer-Encoding: chunked
x-bce-request-id: 6bae5cb3-97d1-4b1a-b8b6-0ad577c1d481
Cache-Control: no-cache
Server: BWS
Date: Tue, 21 May 2021 21:21:21 GMT
Content-Type: application/json;charset=UTF-8

{
    "state": "ACTIVE",
    "presetType": "CUSTOM",
    "createTime": "2021-05-20T21:13:00Z",
    "presetName": "customlized_video_defect_detect_preset",
    "description": "An example video_defect_detect preset description",
    "models": {
        "tooBright": {
            "enable": true,
            "interval": 1000,
            "threshold": 0.5,
            "duration": 2000
        },
        "reddish": {
            "enable": true,
            "interval": 1000,
            "threshold": 0.6,
            "duration": 2000
        },
        "mosaic": {
            "enable": true,
            "interval": 1000,
            "duration": 2000
        }
    }
}
```

查询当前用户及系统的所有模板

接口描述

用户查询其名下及系统提供的所有的模板，具体有哪些系统模板可以参考系统内置模板。

请求（Request）

- 请求语法：

```
GET /v{version}/preset/video_defect_detect HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
connection: keep-alive
accept: */*
host: media.bj.baidubce.com
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

- 请求头域：无特殊Header参数
- 请求参数：无
- 请求体：无

- 请求示例：

```
GET /v3/preset/video_defect_detect HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2021-05-21T21:31:21Z
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: 3807ce30-5264-45f2-9b52-26b78e24a750
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2021-05-21T21:31:21Z/1800/host;x-bce-date/3e1bf9f50ae1fca2d704d61567810dde946fff3ca2e455676455a6f5c8cce596
```

响应（Response）

- 响应头域：无特殊Header参数
- 响应参数：无
- 响应体：与[创建视频质检模板/请求/请求体]保持一致，增加以下字段

字段名称	字段类型	字段描述
presets	List[VddPreset]	模板列表

VddPreset

与[查询指定视频质检模板/响应/响应体]保持一致

- 响应示例：

```
HTTP/1.1 200 OK
Transfer-Encoding: chunked
x-bce-request-id: 3807ce30-5264-45f2-9b52-26b78e24a750
Cache-Control: no-cache
Server: BWS
Date: Tue, 24 Mar 2021 21:31:21 GMT
Content-Type: application/json;charset=UTF-8

{
    "presets": [
        {
            "state": "ACTIVE",
            "presetType": "CUSTOM",
            "createTime": "2021-05-20T21:13:00Z",
            "presetName": "customlized_video_defect_detect_preset",
            "description": "An example video_defect_detect preset description",
            "models": {
                "tooBright": {
                    "enable": true,
                    "interval": 1000,
                    "threshold": 0.5,
                    "duration": 2000
                },
                "reddish": {
                    "enable": true,
                    "interval": 1000,
                    "threshold": 0.6,
                    "duration": 2000
                },
                "mosaic": {
                    "enable": true,
                    "interval": 1000,
                    "duration": 2000
                }
            }
        },
        {
            "state": "ACTIVE",
            "presetType": "CUSTOM",
            "createTime": "2021-05-20T20:14:00Z",
            "presetName": "customlized_video_defect_detect_preset2",
            "description": "An example video_defect_detect preset description 2",
            "models": {
                "tooBright": {
                    "enable": true,
                    "interval": 1000,
                    "threshold": 0.5,
                    "duration": 2000
                }
            }
        }
    ]
}
```

更新视频质检模板

接口描述

通过presetName更新视频质检模板。

请求（Request）

- 请求语法：

```
PUT /v{version}/preset/video_defect_detect/{presetName} HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

- 请求头域：无特殊Header参数

- 请求参数：

字段名称	字段类型	必要性	字段描述	可选值	默认值
presetName	String	可选	任务的模板名称	-	-

- 请求体：同[创建视频质检模板/请求/请求体]

- 请求示例：

```
PUT /v3/preset/video_defect_detect/customlized_video_defect_detect_preset HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2021-05-21T21:31:21Z
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: 0969b91e-1890-470b-b93b-97939f672e54
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2020-02-14T15:12:37Z/1800/host;x-bce-date/7e21c9cf1e4e2cc6921a407a388fe98df122c53b9f509043d841be76eb09a1f9

{
    "presetName": "customlized_video_defect_detect_preset",
    "description": "An example video_defect_detect preset description",
    "models": {
        "tooBright": {
            "enable": true,
            "interval": 1000,
            "threshold": 0.5,
            "duration": 2000
        },
        "reddish": {
            "enable": true,
            "interval": 1000,
            "threshold": 0.5,
            "duration": 2000
        },
        "mosaic": {
            "enable": true,
            "interval": 1000,
            "duration": 2000
        }
    }
}
```

响应（Reponse）

- 响应头域：无特殊Header参数
- 响应参数：无
- 响应体：无
- 响应示例：

```
HTTP/1.1 200 OK
Transfer-Encoding: chunked
x-bce-request-id: 6bae5cb3-97d1-4b1a-b8b6-0ad577c1d481
Cache-Control: no-cache
Server: BWS
Date: Tue, 21 May 2021 21:31:22 GMT
Content-Type: application/json;charset=UTF-8
```

删除视频质检模板

接口描述

通过presetName删除视频质检模板。

请求（Request）

- 请求语法：

```
DELETE /v{version}/preset/video_defect_detect/{presetName} HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

- 请求头域：无特殊Header参数
- 请求参数：

字段名称	字段类型	必要性	字段描述	可选值	默认值
presetName	String	可选	任务的模板名称	-	-

- 请求体：无
- 请求示例：

```
DELETE /v{version}/preset/video_defect_detect/customlized_video_defect_detect_preset HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2021-05-21T21:41:21Z
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: 6bae5cb3-97d1-4b1a-b8b6-0ad577c1d481
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2021-05-21T21:21:21Z/1800/host;x-bce-
date/7e21c9cf1e4e2cc6921a407a388fe98df122c53b9f509043d841be76eb09a1f9
```

响应（Reponse）

- 响应头域：无特殊Header参数
- 响应参数：无
- 响应体：无
- 响应示例：

```
HTTP/1.1 200 OK
x-bce-request-id: 6d0b0a36-2ffe-49d4-9d81-333a9ab9417e
Cache-Control: no-cache
```

视频质量检测任务接口

创建视频质检任务

接口描述

用户通过BOS Bucket、Key以及其他配置信息为指定视频做质量检测。

请求（Request）

- 请求语法：

```
POST /v{version}/job/video_defect_detect HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
connection: keep-alive
accept: */*
host: media.bj.baidubce.com
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

- 请求头域：无特殊Header参数
- 请求参数：无
- 请求体：

字段名称	字段类型	必要性	字段描述	可选值	默认值
pipelineName	String	必选	任务所属的pipelineName	-	-
presetName	String	可选	任务的模板名称，presetName和models参数设置一个即可，推荐设置presetName	-	-
source	Object	必选	输入视频信息的集合	-	-
+ bucket	String	可选	输入视频文件的BOS Bucket（用户必须有该bucket的读权限）	-	队列中指定的输入bucket
+ key	String	必选	输入视频的BOS的Key	-	-
needTarget	Boolean	可选	是否需要出输出检测异常帧	true、false	false
target	Object	可选	检测异常帧的集合	-	-
+ targetBucket	String	可选	检测异常帧的BOS Bucket	-	队列中指定的输出bucket
+ targetFolder	String	可选	存储检测异常帧的目标文件夹名称	-	系统根据source中的key和presetName自动生成
models	ModelsParm	可选	需要检测的项目（使用的模型），models和presetName参数设置一个即可，都设置则该参数会自动覆盖模板中的参数。若设置了models，则ModelsParm中应至少设置一个模型。	-	-
notification	String	可选	通知名称	-	队列中指定的notification

ModelsParm

字段名称	字段类型	必要性	字段描述	可选值	默认值
whiteScreen	ModelParm	可选	白屏检测模型	-	-
blackScreen	ModelParm	可选	黑屏检测模型	-	-
tooBright	ModelParm	可选	过亮检测模型	-	-
tooDark	ModelParm	可选	过暗检测模型	-	-
redScreen	ModelParm	可选	红屏检测模型	-	-
yellowScreen	ModelParm	可选	黄屏检测模型	-	-
greenScreen	ModelParm	可选	绿屏检测模型	-	-
blueScreen	ModelParm	可选	蓝屏检测模型	-	-
purpleScreen	ModelParm	可选	紫屏检测模型	-	-
reddish	ModelParm	可选	偏红检测模型	-	-
yellowish	ModelParm	可选	偏黄检测模型	-	-
greenish	ModelParm	可选	偏绿检测模型	-	-
bluish	ModelParm	可选	偏蓝检测模型	-	-
purplish	ModelParm	可选	偏紫检测模型	-	-
blur	ModelParm	可选	模糊检测模型	-	-
noise	ModelParm	可选	噪声检测模型	-	-
mosaic	ModelParm	可选	马赛克检测模型	-	-
freeze	ModelParm	可选	冻结检测模型	-	-
jitter	ModelParm	可选	抖动检测模型	-	-
blackEdge	ModelParm	可选	黑边检测模型	-	-
blurEdge	ModelParm	可选	模糊边缘检测模型	-	-
staticEdge	ModelParm	可选	静态边缘检测模型	-	-
crash	ModelParm	可选	花屏检测模型	-	-
colorBar	ModelParm	可选	彩条检测模型	-	-
block	ModelParm	可选	块效应检测模型	-	-
interlace	ModelParm	可选	场效应检测模型	-	-
mute	ModelParm	可选	静音检测模型	-	-
volumeLow	ModelParm	可选	音量过低检测模型	-	-
volumeHigh	ModelParm	可选	音量过高检测模型	-	-
soundIntermittent	ModelParm	可选	声音间断检测模型	-	-

ModelParm

字段名称	字段类型	必要性	字段描述	可选值	默认值
enable	Boolean	可选	是否使用该模型	true、false	true
interval	Number	可选	检测间隔（毫秒）	正整数	jitter模型为200，mute/volumeLow/volumeHigh/soundIntermittent模型为100，其他模型均为1000
threshold	Number	可选	检测分值的阈值，对应单帧分值>=threshold时，认为单帧异常。tooBright/tooDark/reddish/yellowish/greenish/bluish/purplish/blur/noise/mosaic/blackEdge/blurEdge/staticEdge/block/volumeLow/volumeHigh/soundIntermittent模型支持阈值自定义，其他模型不支持	0.0~1.0	tooBright/tooDark/blur/noise/reddish/yellowish/greenish/bluish/purplish模型为0.5，mosaic模型为0.001，blackEdge/blurEdge/staticEdge模型为0.2, block模型为0.4, mute/soundIntermittent模型为0.0001, volumeLow模型为0.01, volumeHigh模型为0.6
duration	Number	可选	持续时长（毫秒）的阈值，对应单帧异常持续时长>=duration时，将对应异常时间段写入检测结果中。	非负整数	blur/noise/freeze/volumeLow模型为2000，soundIntermittent模型为100，volumeHigh模型为200，其他模型均为1000

各模型threshold阈值含义

模型	模型描述	默认阈值	阈值含义
whiteScreen	白屏检测模型	-	-
blackScreen	黑屏检测模型	-	-
tooBright	过亮检测模型	0.500	• 表示检测每帧图像过亮程度（值越大程度越大）的阈值，大于阈值认为单帧存在该异常； • 支持阈值自定义[0,1]，阈值越大，表示检测门槛越高。
tooDark	过暗检测模型	0.500	• 表示检测每帧图像过暗程度（值越大程度越大）的阈值，大于阈值认为单帧存在该异常； • 支持阈值自定义，阈值越大，表示检测门槛越高。
redScreen	红屏检测模型	-	-
yellowScreen	黄屏检测模型	-	-
greenScreen	绿屏检测模型	-	-
blueScreen	蓝屏检测模型	-	-
purpleScree	紫屏检测模	-	-

n	型		
reddish	偏红检测模型	0.500	• 表示检测每帧图像偏红程度（值越大程度越大）的阈值，大于阈值认为单帧存在该异常； • 支持阈值自定义[0,1]，设定阈值越大，表示检测门槛越高。
yellowish	偏黄检测模型	0.500	• 表示检测每帧图像偏黄程度（值越大程度越大）的阈值，大于阈值认为单帧存在该异常； • 支持阈值自定义[0,1]，设定阈值越大，表示检测门槛越高。
greenish	偏绿检测模型	0.500	• 表示检测每帧图像偏绿程度（值越大程度越大）的阈值，大于阈值认为单帧存在该异常； • 支持阈值自定义[0,1]，设定阈值越大，表示检测门槛越高。
bluish	偏蓝检测模型	0.500	• 表示检测每帧图像偏蓝程度（值越大程度越大）的阈值，大于阈值认为单帧存在该异常； • 支持阈值自定义[0,1]，设定阈值越大，表示检测门槛越高。
purplish	偏紫检测模型	0.500	• 表示检测每帧图像偏紫程度（值越大程度越大）的阈值，大于阈值认为单帧存在该异常； • 支持阈值自定义[0,1]，设定阈值越大，表示检测门槛越高。
blur	模糊检测模型	0.500	• 表示检测每帧图像模糊程度（值越大程度越大）的阈值，大于阈值认为单帧存在该异常； • 支持阈值自定义[0,1]，设定阈值越大，表示检测门槛越高。
noise	噪声检测模型	0.500	• 表示检测每帧图像噪声程度（值越大程度越大）的阈值，大于阈值认为单帧存在该异常； • 支持阈值自定义[0,1]，设定阈值越大，表示检测门槛越高。
mosaic	马赛克检测模型	0.001	• 表示检测每帧图像中马赛克区域与整个图像面积比值的阈值，大于阈值认为单帧存在该异常； • 支持阈值自定义[0,1]，设定阈值越大，表示检测门槛越高。
freeze	冻结检测模型	-	-
jitter	抖动检测模型	-	-
blackEdge	黑边检测模型	0.200	• 表示检测每帧图像中黑边区域与整个图像面积比值的阈值，大于阈值认为单帧存在该异常； • 支持阈值自定义[0,1]，设定阈值越大，表示检测门槛越高。

blurEdge	模糊边缘检测模型	0.200	• 表示检测每帧图像中模糊边缘区域与整个图像面积比值的阈值，大于阈值认为单帧存在该异常； • 支持阈值自定义[0,1]，设定阈值越大，表示检测门槛越高。
staticEdge	静态边缘检测模型	0.200	• 表示检测每帧图像中静态边缘区域与整个图像面积比值的阈值，大于阈值认为单帧存在该异常； • 支持阈值自定义[0,1]，设定阈值越大，表示检测门槛越高。
crash	花屏检测模型	-	-
colorBar	彩条检测模型	-	-
block	块效应检测模型	0.400	• 表示检测每帧图像块效应程度（值越大程度越大）的阈值，大于阈值认为单帧存在该异常； • 支持阈值自定义[0,1]，设定阈值越大，表示检测门槛越高。
interlace	场效应检测模型	-	-
mute	静音检测模型	0.0001	• 表示检测的每帧音频的音量均值的阈值，小于阈值认为单帧存在该异常； • 支持阈值自定义[0,1]，设定阈值越大，表示检测门槛越高。
volumeLow	音量过低检测模型	0.001	• 表示检测的每帧音频的音量均值的阈值，小于阈值认为单帧存在该异常； • 支持阈值自定义[0,1]，设定阈值越大，表示检测门槛越高。
volumeHigh	音量过高检测模型	0.600	• 表示检测的每帧音频的音量均值的阈值，大于阈值认为单帧存在该异常； • 支持阈值自定义[0,1]，设定阈值越大，表示检测门槛越高。
soundIntermittent	声音间断检测模型	0.0001	• 表示检测的每帧音频的音量均值的阈值，小于阈值认为单帧存在静音，5秒内出现两次及以上静音，则认为存在该异常； • 支持阈值自定义[0,1]，设定阈值越大，表示检测门槛越高。

- 请求示例

```
POST /v3/job/video_defect_detect HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2021-05-21T21:13:00Z
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: d97c57d0-ca44-4d1c-bfeb-941a92440968
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2021-05-21T21:13:00Z/1800/host;x-bce-date/7e21c9cf1e4e2cc6921a407a388fe98df122c53b9f509043d841be76eb09a1f9

{
    "pipelineName": "high_priority_pipe",
    "source": {
        "key": "samplefolderpath/samplevideo.mp4"
    },
    "needTarget": true,
    "target": {
        "targetFolder": "sampleoutputfolderpath"
    },
    "models": {
        "tooBright": {
            "enable": true,
            "interval": 1000,
            "threshold": 0.5,
            "duration": 2000
        },
        "reddish": {
            "enable": true,
            "interval": 1000,
            "threshold": 0.6,
            "duration": 2000
        },
        "mosaic": {
            "enable": true,
            "interval": 1000,
            "duration": 2000
        }
    }
}
```

响应（Reponse）

- 响应头域：无特殊Header参数
- 响应参数：无
- 响应体：

字段名称	字段类型	字段描述
jobId	String	系统生成的任务的唯一标识

- 响应示例：

```
HTTP/1.1 200 OK
Transfer-Encoding: chunked
x-bce-request-id: d97c57d0-ca44-4d1c-bfeb-941a92440968
Cache-Control: no-cache
Server: BWS
Date: Tue, 21 May 2021 21:13:02 GMT
Content-Type: application/json;charset=UTF-8

{
    "jobId":"job-lsdspxdastsmnbam"
}
```

查询指定视频质检任务

接口描述

通过jobId查询视频质检任务。

请求（Request）

- 请求语法：

```
GET /v{version}/job/video_defect_detect/{jobId} HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

- 请求头域：无特殊Header参数
- 请求参数：

字段名称	字段类型	必要性	字段描述	可选值	默认值
jobId	String	必选	系统生成的任务的唯一标识	-	-

- 请求体：无
- 请求示例：

```
GET /v3/job/video_defect_detect/job-lsdspxdastsmnbam HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2021-05-21T21:21:21Z
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: 6bae5cb3-97d1-4b1a-b8b6-0ad577c1d481
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2021-05-21T21:21:21Z/1800/host;x-bce-
date/7e21c9cf1e4e2cc6921a407a388fe98df122c53b9f509043d841be76eb09a1f9
```

响应（Reponse）

- 响应头域：无特殊Header参数
- 响应参数：无

- 响应体：与[创建视频质检任务/请求/请求体]保持一致，增加以下字段

字段名称	字段类型	字段描述
jobId	String	任务的唯一标识
jobStatus	String	任务状态，success/failed/pending/running
usedTime	Number	检测用时（毫秒、ms）
result	ModelsResult	结果
error	Object	错误，可能无此字段
+ code	String	错误码
+ message	String	错误信息
createTime	String	任务创建的时间
startTime	String	任务开始的时间
endTime	String	任务完成的时间

ModelsResult

字段名称	字段类型	字段描述
whiteScreen	ModelResult	白屏检测结果
blackScreen	ModelResult	黑屏检测结果
tooBright	ModelResult	过亮检测结果
tooDark	ModelResult	过暗检测结果
redScreen	ModelResult	红屏检测结果
yellowScreen	ModelResult	黄屏检测结果
greenScreen	ModelResult	绿屏检测结果
blueScreen	ModelResult	蓝屏检测结果
purpleScreen	ModelResult	紫屏检测结果
reddish	ModelResult	偏红检测结果
yellowish	ModelResult	偏黄检测结果
greenish	ModelResult	偏绿检测结果
bluish	ModelResult	偏蓝检测结果
purplish	ModelResult	偏紫检测结果
blur	ModelResult	模糊检测结果
noise	ModelResult	噪声检测结果
mosaic	ModelResult	马赛克检测结果
freeze	ModelResult	冻结检测结果
jitter	ModelResult	抖动检测结果
blackEdge	ModelResult	黑边检测结果
blurEdge	ModelResult	模糊边缘检测结果
staticEdge	ModelResult	静态边缘检测结果
crash	ModelResult	花屏检测结果
colorBar	ModelResult	彩条检测结果
block	ModelResult	块效应检测结果
interlace	ModelResult	场效应检测结果
mute	ModelResult	静音检测结果
volumeLow	ModelResult	音量过低检测结果
volumeHigh	ModelResult	音量过高检测结果
intermittent	ModelResult	声音间断检测结果

ModelResult

字段名称	字段类型	字段描述
success	Boolean	该项目是否成功
defects	Array[DefectResult]	缺陷结果的集合（按时间段分割）
error	Object	错误
+ code	String	错误码
+ message	String	错误信息

DefectResult

字段名称	字段类型	字段描述
score	Number	缺陷分值，检测结果对应时间段内的均值，含义参考模型阈值的含义，tooBright/tooDark/reddish/yellowish/greenish/bluish/purplish/blur/noise/mosaic/blackEdge/blurEdge/staticEdge/block/mute/volumeLow/volumeHigh/soundIntermittent模型有此字段，其他模型无
start	Number	缺陷起始时间戳（毫秒、ms）
end	Number	缺陷结束时间戳（毫秒、ms）
targetKeys	Array[String]	检测异常帧（文件）的BOS key列表，needTarget为true时有此字段，mute/volumeLow/volumeHigh/soundIntermittent模型无此字段，其他模型有

- 响应示例：

```
HTTP/1.1 200 OK
Transfer-Encoding: chunked
x-bce-request-id: 6bae5cb3-97d1-4b1a-b8b6-0ad577c1d481
Cache-Control: no-cache
Server: BWS
Date: Tue, 21 May 2021 21:21:21 GMT
Content-Type: application/json;charset=UTF-8

{
    "jobId": "job-lsdspxdastsmnbwx",
    "jobStatus": "success",
    "pipelineName": "high_priority_pipe",
    "source": {
        "key": "samplefolderpath/samplevideo.mp4"
    },
    "needTarget": true,
    "target": {
        "targetFolder": "sampleoutputfolderpath"
    },
    "usedTime": 12000,
    "result": {
        "tooBright": {
            "success": true,
            "defects": [
                {
                    "score": 0.95,
                    "start": 4000,
                    "end": 5000,
                    "targetKeys": [
                        "sampleoutputfolderpath/00004000.jpg",
                        "sampleoutputfolderpath/00005000.jpg"
                    ]
                }
            ]
        },
        "reddish": {
            "success": true,
            "defects": [
                {
                    "score": 0.95,
                    "start": 9000,
                    "end": 10000,
                    "targetKeys": [
                        "sampleoutputfolderpath/00009000.jpg",
                        "sampleoutputfolderpath/00010000.jpg"
```

```
                    ]
                }
            ]
        },
        "mosaic": {
            "success": true,
            "defects": [
                {
                    "start": 16000,
                    "end": 17000,
                    "targetKeys": [
                        "sampleoutputfolderpath/00016000.jpg",
                        "sampleoutputfolderpath/00017000.jpg"
                    ]
                },
                {
                    "start": 19000,
                    "end": 20000,
                    "targetKeys": [
                        "sampleoutputfolderpath/00019000.jpg",
                        "sampleoutputfolderpath/00020000.jpg"
                    ]
                }
            ]
        }
    },
    "createTime": "2021-05-20T21:13:00Z",
    "startTime" : "2021-05-20T21:13:01Z",
    "endTime": "2021-05-20T21:13:14Z"
}
```


查询指定队列的视频质检任务

接口描述

查询指定队列下满足一定条件的所有质检任务。

请求（Request）

- 请求语法：

```
GET /v{version}/job/video_defect_detect?pipelineName={pipelineName}&jobStatus={jobStatus}&begin={begin}&end={end}}&marker={marker}&maxSize={maxSize} HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

- 请求头域：无特殊Header参数
- 请求参数：

字段名称	字段类型	必要性	字段描述	可选值	默认值
pipelineName	String	必选	任务所属的队列名	-	-
jobStatus	String	可选	所选任务的状态	SUCCESS, FAILED, PENDING, RUNNING	-
begin	String	可选	任务创建时间的上限。所选任务开始时间要大于等于begin	-	-
end	String	可选	任务创建时间的下限，所选任务开始时间要小于等于end	-	-
marker	String	可选	本次请求的marker，标记查询的起始位置，此处为jobId	-	-
maxSize	Number	可选	本次请求返回的任务列表的最大元素个数	1 ~ 1000	1000

- 请求体：无
- 请求示例：

```
GET /v3/job/video_defect_detect?pipelineName=high_priority_pipe&jobStatus=SUCCESS&begin=2021-05-20T08%3A53%3A42Z&end=2021-05-22T08%3A53%3A42Z&marker=job-feumm9etdd5c9gqv&maxSize=2 HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2021-05-21T21:14:00Z
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: 3807ce30-5264-45f2-9b52-26b78e24a750
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2021-05-21T21:14:00Z/1800/host;x-bce-date/3e1bf9f50ae1fca2d704d61567810dde946fff3ca2e455676455a6f5c8cce596
```

响应（Response）

- 响应头域：无特殊Header参数
- 响应参数：无
- 响应体：与[查询指定视频质检任务/响应/响应体]保持一致，增加以下字段

字段名称	字段类型	字段描述
marker	String	本次请求的marker，标记查询的起始位置，此处为jobId
isTruncated	Boolean	指明返回数据是否被截断。true表示本页后面还有数据，即数据未全部返回；false表示已是最后一页，即数据已全部返回
nextMarker	String	获取下一页所需要传递的marker值（此处为jobId），仅当isTruncated为true时（数据未全部返回）出现

- 响应示例：

```
HTTP/1.1 200 OK
Transfer-Encoding: chunked
x-bce-request-id: 6bae5cb3-97d1-4b1a-b8b6-0ad577c1d481
Cache-Control: no-cache
Server: BWS
Date: Tue, 21 May 2021 21:21:21 GMT
```

Content-Type: application/json;charset=UTF-8


```
{
    "jobs" : [
        {
            "jobId": "job-lsdspxdastsmnbwx",
            "jobStatus": "success",
            "pipelineName": "high_priority_pipe",
            "source": {
                "key": "samplefolderpath/samplevideo.mp4"
            },
            "needTarget": true,
            "target": {
                "targetFolder": "sampleoutputfolderpath"
            },
            "usedTime": 12000,
            "result": {
                "tooBright": {
                    "success": true,
                    "defects": [
                        {
                            "score": 0.95,
                            "start": 4000,
                            "end": 5000,
                            "targetKeys": [
                                "sampleoutputfolderpath/00004000.jpg",
                                "sampleoutputfolderpath/00005000.jpg"
                            ]
                        }
                    ]
                },
                "reddish": {
                    "success": true,
                    "defects": [
                        {
                            "score": 0.95,
                            "start": 9000,
                            "end": 10000,
                            "targetKeys": [
                                "sampleoutputfolderpath/00009000.jpg",
                                "sampleoutputfolderpath/00010000.jpg"
                            ]
                        }
                    ]
                },
                "mosaic": {
                    "success": true,
                    "defects": [
                        {
                            "start": 16000,
                            "end": 17000,
                            "targetKeys": [
                                "sampleoutputfolderpath/00016000.jpg",
                                "sampleoutputfolderpath/00017000.jpg"
                            ]
                        },
                        {
                            "start": 19000,
                            "end": 20000,
                            "targetKeys": [
                                "sampleoutputfolderpath/00019000.jpg",
                                "sampleoutputfolderpath/00020000.jpg"
```

```
                    ]
                }
            ]
        }
    },
    "createTime": "2021-05-20T21:13:00Z",
    "startTime" : "2021-05-20T21:13:01Z",
    "endTime": "2021-05-20T21:13:14Z"
},
{
    "jobId": "job-feumm9etdd5c9gqv",
    "jobStatus": "success",
    "pipelineName": "high_priority_pipe",
    "source": {
        "key": "samplefolderpath/samplevideo2.mp4"
    },
    "needTarget": true,
    "target": {
        "targetFolder": "sampleoutputfolderpath"
    },
    "usedTime": 12000,
    "result": {
        "tooBright": {
            "success": true,
            "defects": [
                {
                    "score": 0.95,
                    "start": 11000,
                    "end": 12000,
                    "targetKeys": [
                        "sampleoutputfolderpath/00011000.jpg",
                        "sampleoutputfolderpath/00012000.jpg"
                    ]
                }
            ]
        }
    },
    "createTime": "2021-05-20T21:14:00Z",
    "startTime" : "2021-05-20T21:14:01Z",
    "endTime": "2021-05-20T21:14:14Z"
}
],
"marker": "job-lsdspxdastsmnbwx",
"isTruncated": true,
"nextMarker": "job-gfpj59idrpygsjtw"
}
```


重新运行指定视频质检任务

接口描述

通过jobId和presetName重新运行视频质检任务（任务状态需要为成功或失败）。

请求（Request）

- 请求语法：

```
PUT /v{version}/job/video_defect_detect/{jobId}?rerun&presetName={presetName} HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
connection: keep-alive
accept: */*
host: media.bj.baidubce.com
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

- 请求头域：无特殊Header参数
- 请求参数：

字段名称	字段类型	必要性	字段描述	可选值	默认值
jobId	String	必选	系统生成的任务的唯一标识	-	-
rerun	String	必选	重新运行任务接口的标识	空/任意值	-
presetName	String	可选	任务的模板名称	-	原任务的模板名称

- 请求体：无
- 请求示例

```
PUT /v3/job/video_defect_detect/job-lsdspxdastsmnbam?
rerun&presetName=customlized_video_defect_detect_preset HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2021-05-28T21:21:21Z
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: 6bae5cb3-97d1-4b1a-b8b6-0ad577c1d481
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2021-05-28T21:21:21Z/1800/host;x-bce-
date/7e21c9cf1e4e2cc6921a407a388fe98df122c53b9f509043d841be76eb09a1f9
```

响应（Reponse）

- 响应头域：无特殊Header参数
- 响应参数：无
- 响应体：无
- 响应示例：

```
HTTP/1.1 200 OK
Transfer-Encoding: chunked
x-bce-request-id: 6bae5cb3-97d1-4b1a-b8b6-0ad577c1d481
Cache-Control: no-cache
Server: BWS
Date: Tue, 28 May 2021 21:21:21 GMT
Content-Type: application/json;charset=UTF-8
```

视频数字水印接口

视频数字水印模板接口

视频数字水印密钥模板接口

视频数字水印嵌入接口

说明

视频数字水印嵌入功能需通过发起视频转码任务使用。本文档主要介绍本功能相关参数及使用示例，转码相关接口请参考视频转码模板接口和视频转码任务接口。

为使用数字水印嵌入功能，主要有两种配置方法，适用于两种不同的应用场景，可根据自身需求选择：

1. (推荐)在转码任务中配置数字水印。在转码任务中指定数字水印内容、密钥、算法参数等，可配合一般转码模板(转码模板无需配置数字水印)使用，**无需预先创建数字水印模板**。
2. 在转码模板中配置数字水印。若应用场景的数字水印内容及转码配置相对固定，可预先创建数字水印模板，然后在转码模板中配置数字水印模板、密钥、算法参数，无需在任务中再次配置。

注：若同时在转码模板和转码任务中配置了数字水印参数，则转码任务参数优先生效。例如，在转码模板中配置了数字水印模板，同时在转码任务中配置了数字水印文字内容，则实际嵌入水印为任务中的文字。

算法版本说明

当前视频数字水印功能提供3种算法，可通过digitalWmAlgVersion参数选择。各算法的主要特点如下：

算法版本号	支持的水印类型	视频要求	水印容量	主要特点	适用场景
0	图片、文字(仅支持ASCII字符)	嵌入文字时，宽×高>=327680	不超过40字符	早期版本，抗攻击能力较弱，不推荐使用	不推荐使用
1	文字	宽,高>=320(嵌入字符较多时最低分辨率可能增加)	不超过100字节(中文算作3字节)	可一定程度抵抗裁剪、遮挡、剪辑、拼接、转码压缩等攻击，对时域攻击抵抗能力较好	视频时长短、分辨率较高的场景
2	文字	时长>=30s	不超过100字符(中英文不限)	抗攻击能力强，可一定程度抵抗缩放、遮挡、剪辑、拼接、转码压缩等攻击(长视频效果更佳)	视频时长可满足最低要求的场景

在转码任务中配置数字水印

接口描述

在转码任务中添加数字水印及密钥以在转码时嵌入数字水印。

请求体 注：只列出与视频数字水印功能相关的字段。文字与图片水印必须二选一，不可同时配置。

字段名称	字段类型	必要性	字段描述	可选值	默认值
target	Object	必选	输出信息的集合	-	-
+digitalWmTextContent	String	可选	需嵌入的文字内容	字符数<=100, 支持中英文及标点符号(如需嵌入中文请选择v1及以上版本算法)	-
+digitalWmImageBucket	String	可选	(仅算法0支持图片)需嵌入图片的Bos Bucket	-	-
+digitalWmImageKey	String	可选	(仅算法0支持图片)需嵌入图片的Bos Key	-	-
+digitalWmSecretKeyId	String	可选	数字水印密钥模板ID，密钥用于对水印加密嵌入，提取水印需提供正确密钥	需从用户已创建的数字水印密钥模板选择，生效优先级高于转码模板中的密钥	-
+digitalWmAlgVersion	Integer	可选	算法版本号	0 ~ 2	0
+digitalWmStrength	Float	可选	(算法1、2有效)数字水印嵌入强度，对同一算法，强度越高则抗攻击能力越强，隐蔽性越差	0 ~ 1	0.5

请求示例

```
 POST /v3/job HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2023-11-08T21:21:21Z
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: 6bae5cb3-97d1-4b1a-b8b6-0ad577c1d481
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2023-11-08T21:21:21Z/1800/host;x-bce-date/7e21c9cf1e4e2cc6921a407a388fe98df122c53b9f509043d841be76eb09a1f9
{
    "pipelineName": "high_priority_pipe",
    "source": {
        "clips": [
            {
                "bucket": "input",
                "sourceKey": "source.mp4"
            }
        ]
    },
    "target": {
        "bucket": "output",
        "key": "source_embed.mp4",
        "presetName": "x265_preset",
        "digitalWmTextContent": "baidu",
        "digitalWmSecretKeyId": "key-pi0nwwuw45zx0ya8",
        "digitalWmAlgVersion": 2,
        "digitalWmStrength": 0.5
    },
}
```

在转码模板中配置数字水印

接口描述

在转码模板中添加数字水印模板及密钥以在转码时嵌入数字水印。转码模板的创建、修改等功能请参考视频转码模板接口。

请求体 注：只列出与视频数字水印功能相关的字段。

字段名称	字段类型	必要性	字段描述	可选值	默认值
digitalWmId	String	可选	需嵌入的数字水印模板ID	需从用户已创建的数字水印模板选择	-
digitalWmSecretKeyId	String	可选	数字水印密钥模板ID，密钥用于对水印加密嵌入，提取水印需提供正确密钥	需从用户已创建的数字水印密钥模板选择，生效优先级高于转码模板中的密钥	-
digitalWmAlgVersion	Integer	可选	算法版本号	0 ~ 2	0
digitalWmStrength	Double	可选	(算法1、2有效)数字水印嵌入强度，对同一算法，强度越高则抗攻击能力越强，隐蔽性越差	0 ~ 1	0.5

请求示例

```
 POST /v3/preset HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2023-11-08T21:21:21Z
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: 6bae5cb3-97d1-4b1a-b8b6-0ad577c1d481
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2023-11-08T21:21:21Z/1800/host;x-bce-date/7e21c9cf1e4e2cc6921a407a388fe98df122c53b9f509043d841be76eb09a1f9
{
    "presetName": "embed_preset",
    "description": "embed digital watermark",
    "container": "mp4",
    "clip": {
        "startTimeInSecond": 0
    },
    "audio": {
        "bitRateInBps": 256000
    },
    "video": {
        "codec": "h264",
        "codecOptions": {
            "profile": "baseline"
        },
        "bitRateInBps": 1024000,
        "maxFrameRate": 30,
        "maxWidthInPixel": 4096,
        "maxHeightInPixel": 3072,
        "sizingPolicy": "keep",
        "playbackSpeed": 1.5
    },
    "digitalWmId": "dwm-pa4mz6n8i7sefz0p",
    "digitalWmSecretKeyId": "key-pi0nwwuw45zx0ya8",
    "digitalWmAlgVersion": 2,
    "digitalWmStrength": 0.6,
}
```

查询视频数字水印嵌入任务

视频数字水印嵌入任务的查询接口同视频转码任务，请参考视频转码任务接口使用。

视频数字水印提取接口

创建数字水印提取任务

接口描述

用户通过该接口提取视频中的数字水印。

注：务必选择与嵌入数字水印任务相同的算法版本，否则无法正确提取水印内容。算法说明可见视频数字水印嵌入接口。

请求结构

```
POST /v{version}/job/dwmdetect HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
connection: keep-alive
accept: */*
host: media.bj.baidubce.com
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

请求头域

除公共头域外，无其它特殊头域。

请求参数

无

请求体 | 字段名称 | 字段类型 | 必要性 | 字段描述 | 可选值 | 默认值
| ---- | ------ | ---- | ------- | ------- | --- | pipelineName | String | 必选 | 任务所属的队列名称 | - | - | source | Object | 必选 | 输入视频信息集合 | - | - | +bucket | String | 可选 | 输入视频的BOS Bucket（用户必须有该bucket的读权限） | - | 队列中指定的输入bucket | +key | String | 必选 | 输入视频的BOS Key | - | - | target | String | 可选 | 输出图片水印信息(仅提取图片水印时可配置) | - | - | +bucket | String | 可选 | 输出图片的BOS Bucket（用户必须有该bucket的写权限） | - | 队列中指定的输出bucket | +keyPrefix | String | 可选 | 输出图片的BOS Key前缀，输出的完整key为"前缀_dwm_extract.png" | - | 视频文件名 | digitalWmAlgVersion | String | 可选 | 算法版本号 | 0 ~ 2 | 0 | digitalWmType | String | 必选 | 提取水印类型，图片或文字 | "image", "text" | - | digitalWmSecretKeyId | String | 可选 | 数字水印密钥ID，需要与嵌入时密钥一致 | - | -

响应头域

除公共头域外，无其它特殊头域。

响应参数

字段名称	字段类型	字段描述
jobId	String	系统生成的任务的唯一标识

请求示例

提取图片水印

```
 POST /v3/job/dwmdetect HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2023-11-08T21:21:21Z
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: 6bae5cb3-97d1-4b1a-b8b6-0ad577c1d481
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2023-11-08T21:21:21Z/1800/host;x-bce-
date/7e21c9cf1e4e2cc6921a407a388fe98df122c53b9f509043d841be76eb09a1f9
{
    "pipelineName": "high_priority_pipe",
    "source": {
        "bucket": "input",
        "key": "test.mp4"
    },
    "target": {
        "bucket": "output",
        "keyPrefix": "test"
    } ,
    "digitalWmAlgVersion": 0,
    "digitalWmType": "image",
    "digitalWmSecretKeyId": "key-pi0nwwuw45zx0ya8",
}
```

提取文字水印

```
 POST /v3/job/dwmdetect HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2023-11-08T21:21:21Z
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: 6bae5cb3-97d1-4b1a-b8b6-0ad577c1d481
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2023-11-08T21:21:21Z/1800/host;x-bce-
date/7e21c9cf1e4e2cc6921a407a388fe98df122c53b9f509043d841be76eb09a1f9
{
    "pipelineName": "high_priority_pipe",
    "source": {
        "bucket": "input",
        "key": "test.mp4"
    },
    "digitalWmAlgVersion": 2,
    "digitalWmType": "text",
    "digitalWmSecretKeyId": "key-pi0nwwuw45zx0ya8",
}
```

响应示例

```
HTTP/1.1 200 OK
Transfer-Encoding: chunked
x-bce-request-id: d97c57d0-ca44-4d1c-bfeb-941a92440968
Cache-Control: no-cache
Server: BWS
Date: Wed, 08 Nov 2023 07:36:48 GMT
Content-Type: application/json;charset=UTF-8
{
    "jobId": "job-pk7n403picat90kr"
}
```

查询数字水印提取任务

接口描述

用户通过该接口查询视频数字水印提取历史任务信息。

请求结构

```
GET /v{version}/job/dwmdetect/{jobId} HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
connection: keep-alive
accept: */*
host: media.bj.baidubce.com
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

请求头域

除公共头域外，无其它特殊头域。

请求参数

字段名称	字段类型	必要性	字段描述	可选值	默认值
jobId	String	必选	系统生成的任务的唯一标识	-	-

请求体 无

响应头域

除公共头域外，无其它特殊头域。

响应参数

字段名称	字段类型	字段描述
jobId	String	系统生成的任务的唯一标识
pipelineName	String	任务所属的队列名称
jobStatus	String	任务状态(SUCCESS, FAILED)
createTime	String	任务创建时间
startTime	String	任务开始处理时间
endTime	String	任务结束时间
error	Object	错误信息，任务失败时存在
+code	String	错误码
+message	String	错误信息
source	Object	输入视频信息集合
+bucket	String	输入视频的BOS Bucket
+key	String	输入视频的BOS Key
target	Object	输出提取图片水印信息(仅提取图片水印时有该字段)
+bucket	String	输出图片的BOS Bucket（用户必须有该bucket的写权限）
+keyPrefix	String	输出图片的BOS Key前缀
+keys	Array<String>	提取出的水印图片的BOS Key列表，可能有多个
digitalWmAlgVersion	Integer	算法版本号
digitalWmType	String	提取水印类型
digitalWmSecretKeyId	String	数字水印密钥ID
detectedTexts	Array<String>	提取的水印文字

请求示例

```
GET /v3/digitalwatermark/dwm-pa4mz6n8i7sefz0p HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2023-11-08T21:21:21Z
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: 6bae5cb3-97d1-4b1a-b8b6-0ad577c1d481
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2023-11-08T21:21:21Z/1800/host;x-bce-date/7e21c9cf1e4e2cc6921a407a388fe98df122c53b9f509043d841be76eb09a1f9
```

响应示例

提取图片水印

```
HTTP/1.1 200 OK
Transfer-Encoding: chunked
x-bce-request-id: d97c57d0-ca44-4d1c-bfeb-941a92440968
Cache-Control: no-cache
Server: BWS
Date: Wed, 08 Nov 2023 07:36:48 GMT
Content-Type: application/json;charset=UTF-8
{
    "jobId": "job-pk8uyexbbv7ebji9",
    "pipelineName": "high_priority_pipe",
    "jobStatus": "SUCCESS",
    "createTime": "2023-11-09T12:26:17Z",
    "startTime": "2023-11-09T12:26:18Z",
    "endTime": "2023-11-09T12:26:27Z",
    "source": {
        "bucket": "input",
        "key": "test.mp4"
    },
    "target": {
        "bucket": "output",
        "keyPrefix": "test",
        "keys": [
            "test_dwm_extract.png"
        ]
    },
    "digitalWmAlgVersion": 0,
    "digitalWmType": "image",
    "digitalWmSecretKeyId": "key-pi0nwwuw45zx0ya8"
}
```

提取文字水印

```
HTTP/1.1 200 OK
Transfer-Encoding: chunked
x-bce-request-id: d97c57d0-ca44-4d1c-bfeb-941a92440968
Cache-Control: no-cache
Server: BWS
Date: Wed, 08 Nov 2023 07:36:48 GMT
Content-Type: application/json;charset=UTF-8

{
    "jobId": "job-pk8uyexbbv7ebji9",
    "pipelineName": "high_priority_pipe",
    "jobStatus": "SUCCESS",
    "createTime": "2023-11-09T12:26:17Z",
    "startTime": "2023-11-09T12:26:18Z",
    "endTime": "2023-11-09T12:26:27Z",
    "source": {
        "bucket": "input",
        "key": "test.mp4"
    },
    "digitalWmAlgVersion": 2,
    "digitalWmType": "text",
    "digitalWmSecretKeyId": "key-pi0nwwuw45zx0ya8",
    "detectedTexts": [
        "baidu"
    ]
}
```

图片数字水印嵌入接口

创建图片数字水印嵌入任务

接口描述

用户通过该接口向图片中嵌入文字或图片形式的数字水印。图片数字水印服务目前支持两种算法，用户可根据需求选择合适的算法(通过请求参数的algorithmVersion配置)。

算法0（二值图片嵌入算法）：支持嵌入图片和文字，任何水印内容都将以二值图片形式嵌入到载体图片中（文字会自动转换为图片）。因此，无论嵌入图片还是文字，提取水印结果始终为图片，需要人工查看提取结果图片是否存在水印。该算法能够抵抗一定程度的缩放、裁剪、遮挡、截屏等攻击，但抗图片压缩能力较弱，水印容量相对较小。

算法1（文字编码嵌入算法）：仅支持嵌入文字，其将水印以编码形式嵌入，提取结果为字符串。该算法可抵抗一定程度的裁剪、遮挡、截屏、图片压缩等攻击，水印容量大，但不可抵抗缩放攻击。

请求结构

```
POST /v{version}/job/imagedwm HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
connection: keep-alive
accept: */*
host: media.bj.baidubce.com
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

请求头域

除公共头域外，无其它特殊头域。

请求参数

无

请求体

字段名称	字段类型	必要性	字段描述	可选值	默认值
pipelineName	String	必选	任务所属的队列名称	-	-
source	Object	必选	输入图片信息集合(支持BOS、url两种方式，必选其一)	-	-
+bucket	String	可选	输入图片的BOS Bucket（用户必须有该bucket的读权限）	-	队列中指定的输入bucket
+key	String	可选	输入图片的BOS Key	-	-
+url	String	可选	输入图片的url（支持http/https协议）	字符数<1024	-
target	Object	必选	输出配置	-	-
+bucket	String	可选	输出图片的BOS Bucket（用户必须有该bucket的写权限）	-	队列中指定的输出bucket
+key	String	可选	输出图片的BOS Key	-	-
+format	String	可选	输出图片的格式，默认使用BOS Key后缀作为输出格式	bmp、jpg、jpeg、png、webp、tiff	-
+quality	Integer	可选	输出图片的压缩质量分数，越高画质越好，文件体积越大（压缩质量过低可能导致无法提取水印，建议搭配合适的嵌入强度使用）	1 ~ 100	100
digitalWm	Object	必选	嵌入水印信息集合（图片和文字必须二选一）	-	-
+imageBucket	String	可选	水印图片的BOS Bucket（用户必须有该bucket的读权限）	-	-
+imageKey	String	可选	水印图片的BOS Key	-	-
+imageUrl	String	可选	水印图片的url（支持http/https协议）	-	-
+textContent	String	可选	水印文字内容	1~100个字符，支持英文、数字及常用特殊字符	-
taskType	String	必选	任务类型，嵌入任务必填embed	embed	-
algorithmVersion	Integer	可选	算法版本号，算法介绍见文档开头	0，1	0
strength	Double	可选	嵌入强度，越高抗攻击性越强，对画质影响也越大	0 ~ 1	算法0默认为1.0，算法1默认为0.5
secretKey	String	可选	(仅算法1支持)密钥，用于对水印信息加密。提取水印时，需提供与嵌入时一致的密钥才能正确提取出水印内容	1~16个字符，支持英文、数字及常用特殊字符	默认为用户生成唯一密钥用于加密(每个用户仅生成一次且不同用户密钥不同)

响应头域

除公共头域外，无其它特殊头域。

响应参数

字段名称	字段类型	字段描述
jobId	String	系统生成的任务的唯一标识

请求示例

选择算法0嵌入图片水印，使用BOS输入输出，并设置输出图片质量、算法强度等参数。

```
 POST /v3/job/imagedwm HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2023-11-08T21:21:21Z
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: 6bae5cb3-97d1-4b1a-b8b6-0ad577c1d481
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2023-11-08T21:21:21Z/1800/host;x-bce-
date/7e21c9cf1e4e2cc6921a407a388fe98df122c53b9f509043d841be76eb09a1f9
{
    "pipelineName": "high_priority_pipe",
    "source": {
        "bucket": "input",
        "key": "source.jpg"
    },
    "target": {
        "bucket": "output",
        "key": "source_embed.png",
        "quality": 60
    },
    "digitalWm": {
        "imageBucket": "input",
        "imageKey": "wm.png"
    },
    "strength": 0.8,
    "taskType": "embed",
    "algorithmVersion": 0
}
```

选择算法1嵌入文字水印，使用url输入、BOS输出，并设置输出图片质量、算法强度等参数。

```
POST /v3/job/imagedwm HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2023-11-08T21:21:21Z
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: 6bae5cb3-97d1-4b1a-b8b6-0ad577c1d481
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2023-11-08T21:21:21Z/1800/host;x-bce-date/7e21c9cf1e4e2cc6921a407a388fe98df122c53b9f509043d841be76eb09a1f9
{
    "pipelineName": "high_priority_pipe",
    "source": {
        "url": "https://www.baidu.com/source.webp"
    },
    "target": {
        "bucket": "output",
        "key": "source_embed.jpg",
        "quality": 90
    },
    "digitalWm": {
        "textContent": "baidumcp"
    },
    "strength": 0.6,
    "taskType": "embed",
    "algorithmVersion": 1
}
```

响应示例

```
HTTP/1.1 200 OK
Transfer-Encoding: chunked
x-bce-request-id: d97c57d0-ca44-4d1c-bfeb-941a92440968
Cache-Control: no-cache
Server: BWS
Date: Wed, 08 Nov 2023 07:36:48 GMT
Content-Type: application/json;charset=UTF-8
{
    "jobId": "job-pk7n403picat90kr"
}
```

查询图片数字水印嵌入任务

接口描述

用户通过该接口查询图片数字水印嵌入历史任务信息。

请求结构

```
GET /v{version}/job/imagedwm/{jobId} HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
connection: keep-alive
accept: */*
host: media.bj.baidubce.com
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

请求头域

除公共头域外，无其它特殊头域。

请求参数

字段名称	字段类型	必要性	字段描述	可选值	默认值
jobId	String	必选	系统生成的任务的唯一标识	-	-

请求体

无

响应头域

除公共头域外，无其它特殊头域。

响应参数

字段名称	字段类型	字段描述
jobId	String	系统生成的任务的唯一标识
pipelineName	String	任务所属的队列名称
jobStatus	String	任务状态(SUCCESS, FAILED)
createTime	String	任务创建时间
startTime	String	任务开始处理时间
endTime	String	任务结束时间
error	Object	错误信息，任务失败时存在
+code	String	错误码
+message	String	错误信息
source	Object	输入图片信息集合
+bucket	String	输入图片的BOS Bucket
+key	String	输入图片的BOS Key
+url	String	输入图片的url
target	Object	输出配置
+bucket	String	输出图片的BOS Bucket
+key	String	输出图片的BOS Key
+format	String	输出图片的格式
+quality	String	输出图片的质量
digitalWm	Object	嵌入水印信息集合
+imageBucket	String	水印图片的BOS Bucket
+imageKey	String	水印图片的BOS Key
+imageUrl	String	水印图片的url
+textContent	String	水印文字内容
taskType	String	任务类型
algorithmVersion	Integer	算法版本号
strength	Double	嵌入强度
output	Object	嵌入结果信息
+imageBucket	String	输出图片的BOS Bucket（任务成功时与target一致）
+imageKey	String	输出图片的BOS Key（任务成功时与target一致）
taskType	String	任务类型

请求示例

```
 GET /v3/job/imagedwm/job-pk7n403picat90kr HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2023-11-08T21:21:21Z
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: 6bae5cb3-97d1-4b1a-b8b6-0ad577c1d481
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2023-11-08T21:21:21Z/1800/host;x-bce-
date/7e21c9cf1e4e2cc6921a407a388fe98df122c53b9f509043d841be76eb09a1f9
```

响应示例

```
 HTTP/1.1 200 OKTransfer-Encoding: chunked
x-bce-request-id: d97c57d0-ca44-4d1c-bfeb-941a92440968
Cache-Control: no-cache
Server: BWS
Date: Wed, 08 Nov 2023 07:36:48 GMT
Content-Type: application/json;charset=UTF-8
{
    "jobId": "job-pk7n403picat90kr",
    "pipelineName": "high_priority_pipe",
    "jobStatus": "SUCCESS",
    "createTime": "2023-11-08T07:36:45Z",
    "startTime": "2023-11-08T07:36:45Z",
    "endTime": "2023-11-08T07:36:48Z",
    "source": {
        "bucket": "input",
        "key": "source.jpg"
    },
    "target": {
        "bucket": "output",
        "key": "source_embed.png",
        "quality": 60,
        "format": "png"
    },
    "digitalWm": {
        "imageBucket": "input",
        "imageKey": "wm.png"
    },
    "taskType": "embed",
    "algorithmVersion": 0,
    "strength": 0.5,
    "output": {
        "imageBucket": "output",
        "imageKey": "source_embed.png"
    }
}
```

通知接口配置

图片数字水印服务支持配置通知接口接收任务结束消息，避免轮询请求，使用方法同视频转码功能，参考通知接口和队列接口。

图片数字水印提取接口

创建图片数字水印提取任务

接口描述

用户通过该接口从可能嵌入了数字水印的图片中提取水印。图片数字水印服务目前支持两种算法，需选择**与嵌入时相同的算法**才能够正确提取出水印。

算法0（二值图片嵌入算法）：支持嵌入图片和文字，任何水印内容都将以二值图片形式嵌入到载体图片中（文字会自动转换为图片）。因此，无论嵌入图片还是文字，提取水印结果始终为图片，需要人工查看提取结果图片是否存在水印。该算法能够抵抗一定程度的缩放、裁剪、遮挡、截屏等攻击，但抗图片压缩能力较弱，水印容量相对较小。

算法1（文字编码嵌入算法）：仅支持嵌入文字，其将水印以编码形式嵌入，提取结果为字符串。该算法可抵抗一定程度的裁剪、遮挡、截屏、图片压缩等攻击，水印容量大，但不可抵抗缩放攻击。

注：算法0提取结果固定为图片，必须在target字段中配置输出地址。算法1提取结果为字符串，无需配置target。

请求结构

```
POST /v{version}/job/imagedwm HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
connection: keep-alive
accept: */*
host: media.bj.baidubce.com
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

请求头域

除公共头域外，无其它特殊头域。

请求参数

无

请求体

字段名称	字段类型	必要性	字段描述	可选值	默认值
pipelineName	String	必选	任务所属的队列名称	-	-
source	Object	必选	输入图片信息集合	-	-
+bucket	String	可选	输入图片的BOS Bucket（用户必须有该bucket的读权限）	-	队列中指定的输入bucket
+key	String	可选	输入图片的BOS Key	-	-
+url	String	可选	输入图片的url（支持http/https协议）	-	-
target	Object	算法0必选	输出配置（仅限算法0）	-	-
+bucket	String	可选	输出图片的BOS Bucket（用户必须有该bucket的写权限）	-	队列中指定的输出bucket
+key	String	可选	输出图片的BOS Key	-	-
+format	String	可选	输出图片的格式，默认使用BOS Key后缀作为输出格式	bmp、jpg、jpeg、png、webp、tiff	-
+quality	Integer	可选	输出图片的质量，越高则输出文件越大	1 ~ 100	100
taskType	String	必选	任务类型，提取任务必填extract	extract	-
algorithmVersion	Integer	可选	算法版本号	0，1	0
secretKey	String	可选	(仅算法1支持)密钥, 用于对水印信息解密，需提供与嵌入时一致的密钥才能正确提取出水印内容	1~16个字符，支持英文、数字及常用特殊字符	默认使用内置用户级密钥解密，如嵌入时未配置密钥，提取也无需配置

响应头域

除公共头域外，无其它特殊头域。

响应参数

字段名称	字段类型	字段描述
jobId	String	系统生成的任务的唯一标识

请求示例

从使用算法0嵌入过水印的图片中提取水印。

```
POST /v3/job/imagedwm HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2023-11-08T21:21:21Z
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: 6bae5cb3-97d1-4b1a-b8b6-0ad577c1d481
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2023-11-08T21:21:21Z/1800/host;x-bce-date/7e21c9cf1e4e2cc6921a407a388fe98df122c53b9f509043d841be76eb09a1f9
{
    "pipelineName": "high_priority_pipe",
    "source": {
        "bucket": "input",
        "key": "source_embed.png"
    },
    "target": {
        "bucket": "output",
        "key": "source_embed_extract.jpg",
        "quality": 80
    },
    "taskType": "extract",
    "algorithmVersion": 0
}
```

从使用算法1嵌入过水印的图片中提取水印。

```
POST /v3/job/imagedwm HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2023-11-08T21:21:21Z
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: 6bae5cb3-97d1-4b1a-b8b6-0ad577c1d481
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2023-11-08T21:21:21Z/1800/host;x-bce-date/7e21c9cf1e4e2cc6921a407a388fe98df122c53b9f509043d841be76eb09a1f9
{
    "pipelineName": "high_priority_pipe",
    "source": {
        "bucket": "input",
        "key": "source_embed.png"
    },
    "taskType": "extract",
    "algorithmVersion": 1
}
```

响应示例

```
HTTP/1.1 200 OK
Transfer-Encoding: chunked
x-bce-request-id: d97c57d0-ca44-4d1c-bfeb-941a92440968
Cache-Control: no-cache
Server: BWS
Date: Wed, 08 Nov 2023 07:36:48 GMT
Content-Type: application/json;charset=UTF-8
{
    "jobId": "job-pk7n403picat90kr"
}
```

查询图片数字水印提取任务

接口描述

用户通过该接口查询图片数字水印提取历史任务信息。

请求结构

```
GET /v{version}/job/imagedwm/{jobId} HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: {utc-date-string}
connection: keep-alive
accept: */*
host: media.bj.baidubce.com
x-bce-request-id: {bce-request-id}
content-type: application/json
authorization: {bce-authorization-string}
```

请求头域

除公共头域外，无其它特殊头域。

请求参数

字段名称	字段类型	必要性	字段描述	可选值	默认值
jobId	String	必选	系统生成的任务的唯一标识	-	-

请求体

无

响应头域

除公共头域外，无其它特殊头域。

响应参数

字段名称	字段类型	字段描述
jobId	String	系统生成的任务的唯一标识
pipelineName	String	任务所属的队列名称
jobStatus	String	任务状态(SUCCESS, FAILED)
createTime	String	任务创建时间
startTime	String	任务开始处理时间
endTime	String	任务结束时间
error	Object	错误信息，任务失败时存在
+code	String	错误码
+message	String	错误信息
source	Object	输入图片信息集合
+bucket	String	输入图片的BOS Bucket
+key	String	输入图片的BOS Key
+url	String	输入图片的url
target	Object	输出配置（仅限算法0）
+bucket	String	输出图片的BOS Bucket
+key	String	输出图片的BOS Key
+format	String	输出图片的格式
+quality	Integer	输出图片的质量
taskType	String	任务类型
algorithmVersion	Integer	算法版本号
output	Object	提取结果信息
+imageBucket	String	提取结果图片的BOS Bucket（限算法0）
+imageKey	String	提取结果图片的BOS Key（限算法0）
+extractedText	String	提取结果文字（限算法1，若未提取出水印无该字段）

请求示例

```
GET /v3/job/imagedwm/job-pk7n403picat90kr HTTP/1.1
accept-encoding: gzip, deflate
x-bce-date: 2023-11-08T21:21:21Z
host: media.bj.baidubce.com
accept: */*
connection: keep-alive
x-bce-request-id: 6bae5cb3-97d1-4b1a-b8b6-0ad577c1d481
content-type: application/json
authorization: bce-auth-v1/46bd9968a6194b4bbdf0341f2286ccce/2023-11-08T21:21:21Z/1800/host;x-bce-date/7e21c9cf1e4e2cc6921a407a388fe98df122c53b9f509043d841be76eb09a1f9
```

响应示例

使用算法0的提取任务结果。

```
HTTP/1.1 200 OK
Transfer-Encoding: chunked
x-bce-request-id: d97c57d0-ca44-4d1c-bfeb-941a92440968
Cache-Control: no-cache
Server: BWS
Date: Wed, 08 Nov 2023 07:36:48 GMT
Content-Type: application/json;charset=UTF-8
{
    "jobId": "job-pk7n403picat90kr",
    "pipelineName": "high_priority_pipe",
    "jobStatus": "SUCCESS",
    "createTime": "2023-11-08T07:36:45Z",
    "startTime": "2023-11-08T07:36:45Z",
    "endTime": "2023-11-08T07:36:48Z",
    "source": {
        "bucket": "input",
        "key": "source_embed.png"
    },
    "target": {
        "bucket": "output",
        "key": "source_embed_extract.jpg",
        "format": "jpg",
        "quality": 80
    },
    "taskType": "extract",
    "algorithmVersion": 0,
    "output": {
        "imageBucket": "output",
        "imageKey": "source_embed_extract.jpg"
    }
}
```

使用算法1的提取任务结果。

```
HTTP/1.1 200 OK
Transfer-Encoding: chunked
x-bce-request-id: d97c57d0-ca44-4d1c-bfeb-941a92440968
Cache-Control: no-cache
Server: BWS
Date: Wed, 08 Nov 2023 07:36:48 GMT
Content-Type: application/json;charset=UTF-8
{
    "jobId": "job-pk7n403picat90kr",
    "pipelineName": "high_priority_pipe",
    "jobStatus": "SUCCESS",
    "createTime": "2023-11-08T07:36:45Z",
    "startTime": "2023-11-08T07:36:45Z",
    "endTime": "2023-11-08T07:36:48Z",
    "source": {
        "bucket": "input",
        "key": "source_embed.png"
    },
    "taskType": "extract",
    "algorithmVersion": 1,
    "output": {
        "extractedText": "baidumcp"
    }
}
```

通知接口配置

图片数字水印服务支持配置通知接口接收任务结束消息，避免轮询请求，使用方法同视频转码功能，参考通知接口和队列接口。

播放器SDK

播放器SDK文档

百度智能云提供Web、Android及iOS平台的播放器SDK，为开发者提供简单、便捷的开发接口，帮助开发者在各类终端设备上实现媒体播放功能。

- Web平台
  - 播放器Web SDK帮助手册
  - 播放器Web SDK下载
- Android平台
  - 播放器Android SDK帮助手册
  - 播放器Android SDK下载
- iOS平台
  - 播放器iOS SDK帮助手册
  - 播放器iOS SDK下载

服务端SDK

服务端SDK

- Java SDK ：介绍如何使用Java SDK进行任务队列、转码模板、转码任务、水印资源的创建和管理。
- Python SDK ：介绍如何使用Python SDK进行任务队列、转码模板、转码任务、水印资源的创建和管理。
- PHP SDK ：介绍如何使用PHP SDK进行任务队列、转码模板、转码任务、水印资源的创建和管理。
- Golang SDK ：介绍如何使用Golang SDK进行任务队列、转码模板、转码任务、水印资源的创建和管理。

Java-SDK

安装Media-Java-SDK

Media Java SDK目录结构

```
.
com.baidubce
    ├── auth                          //BCE签名相关类
    ├── http                          //BCE的Http通信相关类
    ├── internal                      //SDK内部类
    ├── model                         //BCE公用model类
    ├── services
    │   └── media                     //Media服务相关类
    │       ├── model                 //Media内部model，如Request或Response
    │       └── MediaClient.class     //Media客户端入口类
    ├── util                          //BCE公用工具类
    ├── BceClientConfiguration.class  //对BCE的HttpClient的配置
    ├── BceClientException.class      //BCE客户端的异常类
    ├── BceServiceException.class     //与BCE服务端交互后的异常类
    ├── ErrorCode.class               //BCE通用的错误码
    └── Region.class                  //BCE提供服务的区域
```

直接使用JAR包

步骤如下：

1. 从官方网站下载Java SDK压缩包。

2.将下载的`bce-java-sdk-version.zip`解压后，复制到工程文件夹中。

3.在Eclipse右键“工程 -> Properties -> Java Build Path -> Add JARs”。

4.添加SDK工具包`lib/bce-java-sdk-version.jar`和第三方依赖工具包`third-party/*.jar`。

其中，`version`为版本号，经过上面几步之后，用户就可以在工程中使用Media Java SDK。

运行环境

Java SDK工具包可在jdk1.7、jdk8环境下运行。

版本动态

本文档针对Media Java SDK 0.10.38版本。

快速入门

1.初始化一个MediaClient。

MediaClient是与Media服务交互的客户端，所有Media操作都是通过MediaClient完成的。用户可以参考新建MediaClient，完成初始化客户端的操作。

2.新建一个Pipeline(任务队列)。

通过Pipeline，用户可以更灵活地管理转码任务。当用户创建一个Job(任务)时，用户必须指定一个队列。

用户在创建队列时，需要为队列指定队列的名称、队列的类型（免费型或独享型）、一组源多媒体资源所属的Bucket与目标多媒体资源所属的的Bucket。输入和输出的Bucket可以是不同的。

3.查看Preset(多媒体模板)

查看系统预设的多媒体模板。模板是一个视频资源在做转码计算时所需参数的集合。用户可以更简便的将一个模板应用于一个和多个视频的转码任务，以使这些任务输出相同规格的目标多媒体资源。

音视频转码为用户预设了丰富且完备的系统模板，以满足用户对于目标规格在格式、码率、分辨率、加解密、水印等诸多方向上的普遍需求，对于不希望过多了解音视频复杂技术背景的用户来说，是最佳的选择。

4.新建一个Job(任务)。

创建一个Job执行多媒体转码任务。每个任务将一个原始的多媒体资源转码成目标规格的多媒体资源。因此，任务和转码的目标是一一对应的，也就是说如果用户需要将一个原始视频规格转换成三种目标规格，比如从AVI格式转码成FLV/MP4/HLS格式，那么用户将会需要创建三个任务。

用户在创建任务时，需要为任务指定所属的队列、所需应用的转码模板以及原始音视频资源的BOS Key以及目标音视频资源BOS Key。

5.查询指定Object多媒体格式信息

用户通过BOS Bucket+Key获取指定多媒体文件的媒体信息。

MediaClient

新建MediaClient

MediaClient是Media服务的Java客户端，为调用者与Media服务进行交互提供了一系列的方法。

用户可以参考如下代码新建一个MediaClient：

```
 public class Sample {
public static void main(String[] args) {
    String ACCESS_KEY_ID = "your-access-key-id";
    String SECRET_ACCESS_KEY = "your-secret-access-key";

    // 初始化一个MediaClient
    BceClientConfiguration config = new BceClientConfiguration();
    config.setCredentials(new DefaultBceCredentials(ACCESS_KEY_ID, SECRET_ACCESS_KEY));
    MediaClient client = new MediaClient(config);

    }
}
```

在上面代码中，变量ACCESS_KEY_ID与SECRET_ACCESS_KEY是由系统分配给用户的，均为字符串，用于标识用户，为访问Media做签名验证。其中ACCESS_KEY_ID对应控制台中的“Access Key ID”，SECRET_ACCESS_KEY对应控制台中的“Access Key Secret”，获取方式请参考获取AK/SK。

上面的方式使用默认域名作为Media的服务地址，如果用户需要自己制定域名，可以通过传入ENDPOINT参数来指定。

```
 String ACCESS_KEY_ID = "your-access-key-id";
String SECRET_ACCESS_KEY = "your-secret-access-key";
String ENDPOINT = "http://media.bj.baidubce.com";

BceClientConfiguration config = new BceClientConfiguration();
config.setCredentials(new DefaultBceCredentials(ACCESS_KEY_ID,SECRET_ACCESS_KEY));
config.setEndpoint(ENDPOINT);
MediaClient client = new MediaClient(config);
```

注意：ENDPOINT参数只能用指定的包含Region的域名来进行定义，目前MCP提供了“华北-北京”、“华南-广州”和“华东-苏州”三个Region。详细的服务域名可以参考：服务域名。 随着Region的增加将会开放其他可以支持的域名。

配置MediaClient

如果用户需要配置MediaClient的一些细节的参数，可以在构造MediaClient的时候传入BceClientConfiguration对象。BceClientConfiguration是Media服务的配置类，可以为客户端配置代理，最大连接数等参数。

使用代理

下面一段代码可以让客户端使用代理访问Media服务：

```
 String ACCESS_KEY_ID = "your-access-key-id";
String SECRET_ACCESS_KEY = "your-secret-access-key";
String ENDPOINT = "http://media.bj.baidubce.com";

// 创建BceClientConfiguration实例
BceClientConfiguration config = new BceClientConfiguration();

// 配置代理为本地8080端口
config.setProxyHost("127.0.0.1");
config.setProxyPort(8080);

// 配置认证秘钥和服务器信息
config.setCredentials(new DefaultBceCredentials(ACCESS_KEY_ID,SECRET_ACCESS_KEY));
config.setEndpoint(ENDPOINT);

// 创建Media客户端
MediaClient client = new MediaClient(config);
```

使用上面的代码段，客户端的所有操作都会通过127.0.0.1地址的8080端口做代理执行。

对于有用户验证的代理，可以通过下面的代码段配置用户名和密码：

```
 // 创建BceClientConfiguration实例
BceClientConfiguration config = new BceClientConfiguration();

// 配置代理为本地8080端口
config.setProxyHost("127.0.0.1");
config.setProxyPort(8080);

//设置用户名和密码
config.setProxyUsername("username");
config.setProxyPassword("password");
```

设置网络参数

用户可以用BceClientConfiguration对基本网络参数进行设置：

```
 BceClientConfiguration config = new BceClientConfiguration();

// 设置HTTP最大连接数为10
config.setMaxConnections(10);

// 设置TCP连接超时为5000毫秒
config.setConnectionTimeoutInMillis(5000);

// 设置Socket传输数据超时的时间为2000毫秒
config.setSocketTimeoutInMillis(2000);
```

BceClientConfiguration参数说明

通过BceClientConfiguration能指定的所有参数如下表所示：

参数	说明
UserAgent	用户代理，指HTTP的User-Agent头
Protocol	连接协议类型
ProxyDomain	访问NTLM验证的代理服务器的Windows域名
ProxyHost	代理服务器主机地址
ProxyPort	代理服务器端口
ProxyUsername	代理服务器验证的用户名
ProxyPassword	代理服务器验证的密码
ProxyPreemptiveAuthenticationEnabled	是否设置用户代理认证
ProxyWorkstation	NTLM代理服务器的Windows工作站名称
LocalAddress	本地地址
ConnectionTimeoutInMillis	建立连接的超时时间（单位：毫秒）
SocketTimeoutInMillis	通过打开的连接传输数据的超时时间（单位：毫秒）
MaxConnections	允许打开的最大HTTP连接数
RetryPolicy	连接重试策略
SocketBufferSizeInBytes	Socket缓冲区大小

Pipeline队列

队列分为免费型与专享型：

- 免费型队列中的转码任务分享百度智能云为音视频转码所提供的约400路720P转码计算资源。
- 专享型队列需额外采购，以便更好的满足那些对于转码时效性和稳定性有更高要求的用户的业务需求。

用户可以利用队列实现任务优先级。用户通过创建多个队列达到区分任务优先级的目的，将大部分任务创建至普通优先级队列，将高优的任务放入高优先级的队列，以利用队列先到先服务的工作原理来实现任务的优先级调整。

新建Pipeline

如下代码可以新建一个Pipeline，默认的capacity值为20：

```
public void createPipeline (MediaClient client, String pipelineName,
    String sourceBucket, String targetBucket, int capacity) {
  // 新建一个Pipeline
  client.createPipeline(pipelineName, sourceBucket, targetBucket, capacity);
}
```

列出全部Pipeline

如下代码可以列出用户所有的Pipeline：

```
public void listPipelines (MediaClient client) {
  // 获取用户的Pipeline列表
  List<PipelineStatus> list = client.listPipelines().getPipelines();

  // 遍历Pipeline列表
  for (PipelineStatus pipeline : list) {
    System.out.println(pipeline);
  }
}
```

查询指定的Pipeline

若只是查询某个Pipeline，则使用如下代码：

```
public void getPipeline (MediaClient client, String pipelineName) {
    GetPipelineResponse pipeline = client.getPipeline(pipelineName);
    System.out.println(pipeline);
}
```

删除Pipeline

如下代码可以删除一个Pipeline：

```
public void deletePipeline (MediaClient client, String pipelineName) {
    // 删除Pipeline
    client.deletePipeline(pipelineName);
}
```

需要注意的是，如果Pipeline有关联的Job未完成，则Pipeline无法被删除，必须等Job执行结束后才能成功删除。

更新指定的Pipeline

更新指定pipeline的capacity：

```
public void updatePipeline (MediaClient client, String pipelineName, Integer capacity) {
    PipelineStatus pipeline = client.getPipeline(pipelineName).getPipeline();
    pipeline.getConfig().setCapacity(capacity);
    client.updatePipeline(new UpdatePipelineRequest().withPipeline(pipeline));
}
```

更新其他字段类似写法

Transcoding-Job转码任务

Transcoding Job(任务)是音视频转码中最基本的执行单元，每个任务将一个原始的音视频资源转码成目标规格的音视频资源。因此，任务和转码的目标是一一对应的，也就是说如果用户需要将一个原始多媒体文件转换成三种目标规格，比如从AVI格式转码成FLV/MP4/HLS格式，那么用户将会需要创建三个任务。

创建Transcoding Job

用户在创建转码任务时，需要为转码任务指定所属的Pipeline、所需应用的Preset以及原始音视频资源的BOS Key以及目标音视频资源BOS Key。

如下代码创建一个Job，并获取新创建的jobID：

```
public void createJob(MediaClient client, String pipelineName,
        String sourceKey, String targetKey, String presetName) {
    CreateTranscodingJobResponse job = client.createTranscodingJob(pipelineName, sourceKey, targetKey, presetName);
    System.out.println("jobId:" + job.getJobId());
}
```

如下代码创建一个支持视频合并、去水印、加水印（Job上而不是Preset上指定watermarkId）的Job，并获取新创建的jobID：

```
public void createJob(MediaClient client, String pipelineName,
        List<SourceClip> clips, String targetKey, String presetName, String watermarkId, Area delogoArea) {
    // Area delogoArea = new Area().withX(10).withY(200).withWidth(100).withHeight(150);
    String jobId = mediaClient.createTranscodingJob(pipelineName, clips, targetKey, presetName,
                    watermarkId, delogoArea).getJobId();
}
```

如下代码创建一个支持视频合并、去水印、加水印、去黑边、插入多样叠加效果（Insert）的Job，并获取新创建的jobID：

```
public void createJob(MediaClient client, String pipelineName,
        List<SourceClip> clips, String targetKey, String presetName, String watermarkId, Area delogoArea,
        Area crop, List<Insert> inserts) {
    // Layout layout = new Layout().withHorizontalAlignment("right").withHorizontalOffsetInPixel(10)
    //                             .withVerticalAlignment("top").withVerticalOffsetInPixel(100);
    // Timeline timeline = new Timeline().withStartTimeInMillisecond(4500).withDurationInMillisecond(3000);
    // Insert insert = new Insert().withBucket("testbucket").withKey("logo.png").withType("image")
    //                             .withLayout(layout).withTimeline(timeline);
    String jobId = mediaClient.createTranscodingJob(pipelineName, clips, targetKey, presetName,
                    watermarkId, delogoArea, crop, inserts).getJobId();
}
```

说明：

- warteramrkId、delogo、crop、inserts等可设置为null，表示不使用功能。
- 如果多个视频不转码只需要合并时，Preset中 transmux 设置为true即可实现。
- 目前仅支持普通队列合并多个视频，加速队列暂不支持。

列出指定Pipeline的所有Transcoding Job

如下代码通过指定pipelineName查询该Pipeline下的所有Job：

```
public void listJobs(MediaClient client, String pipelineName) {

    // 获取指定Pipeline下的所有Job信息
    ListTranscodingJobsResponse listTranscodingJobs = client.listTranscodingJobs(pipelineName);
    List<Job> jobs = listTranscodingJobs.getJobs();
    for (Job job : jobs) {
        System.out.println("jobId: " + job.getJobId());
    }
}
```

listTranscodingJobs方法返回ListTranscodingJobsResponse对象，ListTranscodingJobsResponse对象包含了此次listTranscodingJobs请求的返回结果。用户可以通过ListTranscodingJobsResponse中的getJobs方法获取所有Job的描述信息。

查询指定的Transcoding Job信息

用户可以通过如下代码通过jobId读取某个Job：

```
public void getJob(MediaClient client, String jobId) {
    GetTranscodingJobResponse resp = client.getTranscodingJob(jobId);
    System.out.println("pipelineName: " + resp.getPipelineName());
    System.out.println("sourceKey: " + resp.getSource().getSourceKey());
    System.out.println("targetKey: " + resp.getTarget().getTargetKey());
    System.out.println("jobStatus: " + resp.getJobStatus());
    System.out.println("startTime: " + resp.getStartTime());
    System.out.println("endTime: " + resp.getEndTime());
}
```

Preset模板

模板是系统预设的对于一个视频资源在做转码计算时所需定义的集合。用户可以更简便的将一个模板应用于一个和多个视频的转码任务，以使这些任务输出相同规格的目标视频资源。

音视频转码为用户预设了丰富且完备的系统模板，以满足用户对于目标规格在格式、码率、分辨率、加解密、水印等诸多方向上的普遍需求，对于不希望过多了解音视频复杂技术背景的用户来说，是最佳的选择。百度为那些在音视频技术上有着丰富积累的用户，提供了可定制化的转码模板，以帮助他们满足复杂业务条件下的转码需求。

当用户仅需对于音视频的容器格式做变化时，百度提供Transmux模板帮助用户以秒级的延迟快速完成容器格式的转换，比如从MP4转换成HLS，而保持原音视频的属性不变。

查询当前用户Preset及所有系统Preset

用户可以通过如下代码查询所有的Preset

```
public void listPresets(MediaClient client) {
    ListPresetsResponse resp = mediaClient.listPresets();
    for (GetPresetResponse preset : resp.getPresets()) {
        System.out.println("presetName: " + preset.getPresetName());
        System.out.println("description: " + preset.getDescription());
        System.out.println("container: " + preset.getContainer());
        System.out.println("transmux: " + preset.getTransmux());
    }
}
```

查询指定的Preset信息

用户可以通过如下代码指定的某个Preset

```
public void getPreset(MediaClient client, String presetName) {
    GetPresetResponse preset = client.getPreset(presetName);
    System.out.println("presetName: " + preset.getPresetName());
    System.out.println("description: " + preset.getDescription());
    System.out.println("container: " + preset.getContainer());
    System.out.println("transmux: " + preset.getTransmux());
}
```

创建Preset

如果系统预设的Preset无法满足用户的需求，用户可以自定义自己的Preset。根据不同的转码需求，可以使用不同的接口创建Preset。

创建仅支持容器格式转换的Preset

如下代码创建仅执行容器格式转换Preset

```
public void createPreset(MediaClient client, String presetName, String description, String container) {
    client.createPreset(presetName, description, container);
}
```

创建音频文件的转码Preset，不需要截取片段和加密

如果创建一个不需要截取片段和加密的音频文件转码Preset，可以参考如下代码

```
public void createPreset(MediaClient client, String presetName, String description, String container,
            Audio audio) {
    client.createPreset(presetName, description, container, audio);
}
```

创建音频文件转码Preset，需要设置片段截取属性和加密属性

如果创建一个支持截取片段和加密的音频文件转码Preset，可以参考如下代码

```
public void createPreset(MediaClient client, String presetName, String description, String container,
            Clip clip, Audio audio, Encryption encryption) {
    client.createPreset(presetName, description, container, clip, audio, encryption);
}
```

创建视频文件转码Preset，不需要截取片段、加密和水印属性

如果创建一个不需要截取片段，加密和水印的视频文件转码Preset，可以参考如下代码

```
public void createPreset(MediaClient client, String presetName, String description, String container,
            Audio audio, Video video) {
    client.createPreset(presetName, description, container, audio, video);
}
```

创建视频文件转码Preset，需要设置片段截取、加密和水印属性

如果创建一个需要截取片段，加密和添加水印的视频文件转码Preset，可以参考如下代码

```
public void createPreset(MediaClient client, String presetName, String description, String container,
            Clip clip,Audio audio, Video video, Encryption encryption, String watermarkId) {
    client.createPreset(presetName, description, container, clip, audio, video, encryption, watermarkId);
}
```

创建Preset，指定所有的参数

如果需要定制所有配置参数，可以参考如下代码

```
public void createPreset(MediaClient client, String presetName, String description, String container,
            boolean transmux, Clip clip,Audio audio, Video video, Encryption encryption,
            Watermarks watermarks, TransCfg transCfg, ExtraCfg extraCfg) {
    client.createPreset(presetName, description, container, transmux, clip, audio, video, encryption,
        watermarks, transCfg, extraCfg);
}
```

> **说明：**
>
> - watermarks、transCfg、extraCfg等可设置为null，表示不使用功能。
> - 不可同时设置watermarks和watermarkId。推荐设置watermarks，其支持多水印设置且可以应用更多的水印控制功能。

更新Preset

用户可以根据模板名更新自己创建的模板：

```
public void updatePreset(MediaClient client, String presetName, String description, String container) {
    GetPresetResponse preset = client.getPreset(presetName);
    preset.setDescription(description);
    preset.setContainer(container);
    client.updatePreset(new UpdatePresetRequest().withPreset(preset));
}
```

更新其他参数类似

Mediainfo媒体信息

对于BOS中某个Object，可以通过下面代码获取媒体信息

```
public void getMediaInfoOfFile(MediaClient client, String bucket, String key) {
    GetMediaInfoOfFileResponse mediaInfo = client.getMediaInfoOfFile(bucket, key);
    System.out.println("fileSizeInByte: " + mediaInfo.getFileSizeInByte());
    System.out.println("etag: " + mediaInfo.getEtag());
    System.out.println("type: " + mediaInfo.getType());
}
```

Thumbnail-Job缩略图任务

缩略图是图片、视频经压缩方式处理后的小图。因其小巧，加载速度非常快，故用于快速浏览。缩略图任务可用于为BOS中的多媒体资源创建缩略图。

创建Thumbnail Job

通过pipeline，BOS Key以及其他配置信息为指定媒体生成缩略图，并获取返回的缩略图任务jobId。可以参考如下代码：

```
public void createThumbnailJob(MediaClient client, String pipelineName, String sourceKey) {
    ThumbnailTarget target = new ThumbnailTarget().withFormat("jpg").withSizingPolicy("keep");

    ThumbnailCapture capture =
            new ThumbnailCapture().withMode("manual")
                .withStartTimeInSecondDouble(0.0D)
                .withEndTimeInSecondDouble(5.0D)
                .withIntervalInSecondDouble(1.0D);

    String jobId =
            mediaClient.createThumbnailJob(pipelineName, sourceKey, target, capture).getJobId();

}
```

创建去水印、去黑边的缩略图，可以参考如下代码：

```
public void createThumbnailJob(MediaClient client, String pipelineName, String sourceKey, Area delogoArea,
        Area crop) {
    ThumbnailTarget target = new ThumbnailTarget().withFormat("gif").withSizingPolicy("keep");

    ThumbnailCapture capture =
        new ThumbnailCapture().withMode("split").withFrameNumber(10);
    String jobId =
        mediaClient.createThumbnailJob(pipelineName, sourceKey, target, capture, delogoArea, crop).getJobId();
}
```

查询指定Thumbnail Job

如果需要获取一个已创建的缩略图任务的信息，可以参考如下代码：

```
public void getThumbnailJob(MediaClient client, String jobId) {
    GetThumbnailJobResponse resp = mediaClient.getThumbnailJob(jobId);

    System.out.println("  jobId = " + resp.getJobId());
    System.out.println("  pipelineName = " + resp.getPipelineName());
    System.out.println("  jobStatus = " + resp.getJobStatus());
    System.out.println("  source.key = " + resp.getSource().getKey());
    System.out.println("  target.keyPrefix = " + resp.getTarget().getKeyPrefix());
    System.out.println("  target.format = " + resp.getTarget().getFormat());
    System.out.println("  target.sizingPolicy = " + resp.getTarget().getSizingPolicy());
    System.out.println("  target.heightInPixel = " + resp.getTarget().getHeightInPixel());
    System.out.println("  target.widthInPixel = " + resp.getTarget().getWidthInPixel());
    System.out.println("  target.keys = " + resp.getTarget().getKeys());
    System.out.println("  capture.mode = " + resp.getCapture().getMode());
    System.out.println("  capture.startTimeInSecond = " + resp.getCapture().getStartTimeInSecond());
    System.out.println("  capture.endTimeInSecond = " + resp.getCapture().getEndTimeInSecond());
    System.out.println("  capture.intervalInSecond = " + resp.getCapture().getIntervalInSecond());
    System.out.println("  capture.frameNumber = " + resp.getCapture().getFrameNumber());
}
```

查询指定队列的Thumbnail Jobs

如果需要获取一个队列里的全部缩略图任务的信息，可以参考如下代码：

```
public void listThumbnailJobs(MediaClient client, String pipelineName) {
  ListThumbnailJobsResponse resp = mediaClient.listThumbnailJobs(pipelineName);

  for (ThumbnailJobStatus job : resp.getThumbnails()) {
    System.out.println("  jobId = " + job.getJobId());
    System.out.println("  pipelineName = " + job.getPipelineName());
    System.out.println("  jobStatus = " + job.getJobStatus());
    System.out.println("  source.key = " + job.getSource().getKey());
    System.out.println("  target.keyPrefix = " + job.getTarget().getKeyPrefix());
    System.out.println("  target.format = " + job.getTarget().getFormat());
    System.out.println("  target.sizingPolicy = " + job.getTarget().getSizingPolicy());
    System.out.println("  target.heightInPixel = " + job.getTarget().getHeightInPixel());
    System.out.println("  target.widthInPixel = " + job.getTarget().getWidthInPixel());
    System.out.println("  target.keys = " + job.getTarget().getKeys());
    System.out.println("  capture.mode = " + job.getCapture().getMode());
    System.out.println("  capture.startTimeInSecond = " + job.getCapture().getStartTimeInSecond());
    System.out.println("  capture.endTimeInSecond = " + job.getCapture().getEndTimeInSecond());
    System.out.println("  capture.IntervalInSecond = " + job.getCapture().getIntervalInSecond());
    System.out.println();
  }

}
```

Watermark水印

数字水印是向数据多媒体（如图像、音频、视频信号等）中添加某些数字信息以达到文件真伪鉴别、版权保护等功能。嵌入的水印信息隐藏于宿主文件中，不影响原始文件的可观性和完整性。

用户可以将BOS中的一个Object创建为水印，获得对应的watermarkId。然后在转码任务中将此水印添加到目的多媒体文件。

创建水印

如果需要创建一个水印，指定水印的位置，并获得水印的唯一ID。其中bucket是水印文件所在bucket名称，key是水印文件在该bucket中的文件名。可以参考如下代码：

```
public void createWaterMark(MediaClient client, String bucket, String key) {
  CreateWaterMarkResponse watermark = client.createWaterMark(bucket, key, "left", "top");
  System.out.println("new watermarkId: " + watermark.getWatermarkId());
}
```

如果需要创建一个水印，指定水印的位置、显示时间段、重复显示次数（动态水印）、自动缩放，并获得水印的唯一ID，可以参考如下代码：

```
public void createWaterMark(MediaClient client, String bucket, String key,
        String horizontalAlignment, String verticalAlignment,
        int horizontalOffsetInPixel, int verticalOffsetInPixel,
        Timeline timeline, Integer repeated, Boolean allowScaling) {
  CreateWaterMarkResponse watermark = client.createWaterMark(bucket, key, horizontalAlignment,
    verticalAlignment, timeline, repeated, allowScaling);
  System.out.println("new watermarkId: " + watermark.getWatermarkId());
}
```

接口返回的是包含了watermarkId的一个对象。

创建水印的转码任务

如果需要把水印添加到目标多媒体文件中，可以创建一个带水印的转码Preset，然后使用这个Preset创建转码任务。 可以设置Preset.watermarkId或Preset.watermarks，这里以设置Preset.watermarkId为例。

```
public void createPreset(MediaClient client, String presetName, String description, String container,
        Clip clip,Audio audio, Video video, Encryption encryption, String watermarkId) {

    client.createPreset(presetName, description, container, clip, audio, video, encryption, watermarkId);

    String jobId = client.createJob(pipelineName, sourceKey, targetKey, presetName).getJobId();
}
```

代码参考创建视频文件转码配置和[创建视频文件转码任务](MCT/服务端SDK/Java-SDK/Transcoding-Job转码任务.md#创建Transcoding Job)。

查询指定水印

如果需要查询已创建的水印，可以参考如下代码：

```
public void getWaterMark(MediaClient client, String watermarkId) {
    GetWaterMarkResponse watermark = client.getWaterMark(watermarkId);
    System.out.println("watermarkId: " + watermark.getWatermarkId());
    System.out.println("createTime: " + watermark.getCreateTime());
    System.out.println("bucket: " + watermark.getBucket());
    System.out.println("key: " + watermark.getKey());
    System.out.println("horizontalOffsetInPixel: " + watermark.getHorizontalOffsetInPixel());
    System.out.println("verticalOffsetInPixel: " + watermark.getVerticalOffsetInPixel());
}
```

查询当前用户水印

如果需要查询出本用户所创建的全部水印，可以参考如下代码：

```
public void getWaterMark(MediaClient client) {
    List<WaterMark> watermarks = client.listWaterMark().getWatermarks();
    for (WaterMark watermark : watermarks) {
        System.out.println("watermarkId = " + watermark.getWatermarkId());
        System.out.println("bucket = " + watermark.getBucket());
        System.out.println("key = " + watermark.getKey());
        System.out.println("createTime = " + watermark.getCreateTime());
        System.out.println("horizontalOffsetInPixel = " + watermark.getHorizontalOffsetInPixel());
        System.out.println("VerticalOffsetInPixel = " + watermark.getVerticalOffsetInPixel());
        System.out.println();
    }
}
```

删除水印

如果需要删除某个已知watermarkId的水印，可以参考如下代码：

```
public void getWaterMark(MediaClient client, String watermarkId) {
    client.deleteWaterMark(watermarkId);
}
```

Notification通知

通知功能可以在音视频转码任务状态转换时主动向开发者服务器推送消息。

创建通知

如果需要创建通知可以参考如下代码：

```
  public void createNotification(MediaClient client, String name, String endpoint) {
    client.createNotification(name, endpoint);
}
```

查询指定通知

如果需要查询已创建的通知，可以参考如下代码：

```
  public void getNotification(MediaClient client, String name) {
    GetNotificationResponse notification = client.getNotification(name);
    System.out.println(notification.getName());
    System.out.println(notification.getEndpoint());
}
```

查询当前用户通知

如果需要查询出本用户所创建的全部通知，可以参考如下代码：

```
  public void ListNotification(MediaClient client) {
    List<Notification> notifications = client.listNotification().getNotifications();
    for (Notification notifciation : notifications) {
        System.out.println(notifciation.getName());
        System.out.println(notifciation.getEndpoint());
    }
}
```

删除通知

如果需要删除某个通知，可以参考如下代码：

```
  public void deleteNotification(MediaClient client, String name) {
    client.deleteNotification(name);
}
```

日志

Media Java SDK发布版本中增加了logback作为slf4j的实现，如果用户没有自己的实现可以直接用，如果工程中有其他的如log4j则可以替代。

默认日志

如果用户使用默认的logback，则需要配置logback.xml到classpath中。如果没有这个配置文件，日志级别默认为DEBUG。

```
 <configuration>
   <property name="LOG_HOME" value="./log/"/>
   <appender name="STDOUT" class="ch.qos.logback.core.ConsoleAppender">
      <!-- encoders are assigned the type
           ch.qos.logback.classic.encoder.PatternLayoutEncoder by default -->
      <encoder>
         <pattern>%d{HH:mm:ss.SSS} [%thread] %-5level %logger{36} - %msg%n</pattern>
      </encoder>
   </appender>

   <appender name="FILE" class="ch.qos.logback.core.rolling.RollingFileAppender">
      <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">
         <FileNamePattern>${LOG_HOME}/MediaUnitTest.%d{yyyy-MM-dd}.log</FileNamePattern>
         <MaxHistory>30</MaxHistory>
      </rollingPolicy>
      <encoder class="ch.qos.logback.classic.encoder.PatternLayoutEncoder">
         <pattern>%d{yyyy-MM-dd HH:mm:ss.SSS} [%thread] %-5level %logger{50} - %msg%n</pattern>
      </encoder>
      <triggeringPolicy class="ch.qos.logback.core.rolling.SizeBasedTriggeringPolicy">
         <MaxFileSize>10MB</MaxFileSize>
      </triggeringPolicy>
   </appender>

   <root level="info">
      <appender-ref ref="STDOUT"/>
      <appender-ref ref="FILE"/>
   </root>
</configuration>
```

自有日志模块

若用户使用自己的日志实现模块，例如项目依赖于Maven，则可以类似下面的配置到pom.xml中来去除logback。

```
 <?xml version="1.0" encoding="utf-8"?>

<dependency>
 <groupId>com.baidubce</groupId>
 <artifactId>bce-java-sdk</artifactId>
 <version>${bce.sdk.version}</version>
 <exclusions>
   <exclusion>
     <groupId>ch.qos.logback</groupId>
     <artifactId>logback-classic</artifactId>
   </exclusion>
   <exclusion>
     <groupId>ch.qos.logback</groupId>
     <artifactId>logback-core</artifactId>
   </exclusion>
   <exclusion>
     <groupId>org.slf4j</groupId>
     <artifactId>jcl-over-slf4j</artifactId>
   </exclusion>
 </exclusions>
</dependency>
```

版本更新记录

v0.10.28

- 支持Preset中指定回放速度playbackSpeed，该值低于1.0为减速，高于1.0为加速（不可同时指定音频设置）。

- 支持多视频合并，通过clips指定多个待合并视频。
- 支持在Job中指定水印id（使用watermarkIds数组传入水印id，当前仅支持单个水印）。
- 支持转码、缩略图服务去水印，使用delogoArea指定去水印区域。
- 支持缩略图按指定张数均匀提取，模式为split，唯一参数frameNumber指定均匀选取的张数。

v0.10.38

支持了 2018-10-18 API更新中的所有功能，主要包括：

- 视频转码任务支持黑边裁剪（Job.crop）。
- 视频转码任务支持视频编辑功能（Job.inserts）。在视频中动态叠加音频、视频、图片、字幕、文本水印等。
- 视频转码支持多水印设置（Preset.watermarks）。
- 视频转码模式新增支持2pass, cae（Content Aware Encoding），cae_enhanced。新的转码模式可以实现相同清晰度下节省码率。Preset.transCfg.transMode支持枚举类型twopass, cae, cae_enhanced。
- 视频转码更多控制。增加Preset.extraCfg.watermarkDisableWhitelist设置满足一定条件（如视频为竖屏）时不加水印，增加Preset.extraCfg.segmentDurationInSecond设置分片时长。
- 水印支持更多设置。增加Watermark.timeline控制水印显示时间段，Watermark.repeated设置动态水印是否循环，Watermark.allowScaling参数允许水印根据视频大小自适应缩放。
- 缩略图任务支持黑边裁剪（Thumbnail.crop）。
- SDK中去水印（delogo）类型由DelogoArea转变为Area。

v0.10.95

- 支持通知（Notification）相关接口，包括创建通知、查询指定通知、查询所有通知、删除指定通知
- 支持更新队列（Pipeline）和模板（Preset）接口
- 支持新参数：
  - 视频转码模板支持设置跳过黑帧（Preset.extraCfg.skipBlackFrame）
  - 视频转码支持在视频clip中设置Bucket
  - 视频转码支持自动裁剪黑边（autoCrop）
  - 缩略图转码支持跳过黑帧(Thumbnail.capture.skipBlackFrame)，highlight模式参数设置(Thumbnail.capture.highlightOutputCfg)和雪碧图参数设置(Thumbnail.capture.spriteOutputCfg)

Python-SDK

简介

本文档主要介绍Media Python SDK的安装和使用。在使用本文档前，您需要先了解音视频转码的一些基本知识，并已开通了音视频转码服务。若您还不了解音视频转码，可以参考产品描述和入门指南。

安装Media-SDK-for-Python

环境准备

1. 运行环境

   Python SDK工具包支持在Python 2.7 以上环境运行。

2. 安装pycrypto依赖

   安装SDK之前，需要先执行命令`pip install pycrypto`安装pycrypto依赖。
   如果安装失败，请执行`pip install pycryptodome`

下载和安装

方式一：通过pip安装

您可以通过pip安装的方式将百度智能云Python SDK安装到您的环境中。 联网状态下，在命令行中执行如下命令：

```
pip install bce-python-sdk
```

即可将Python SDK安装到本地。 **方式二：将源码包下载到本地后进行安装**

1. 在开发者资源中心下载Python SDK压缩工具包。
2. 命令行移动到压缩包所在路径，执行如下命令（version替换为包名称中的版本号）：

   ```
   pip install bce-python-sdk-version.zip
   ```

   即可将Python SDK安装到本地。

您也可以解压压缩包后执行如下命令（version替换为包名称中的版本号）

```
cd bce-python-sdk-version

python setup.py install
```

关于配置文件的引用请参考下文中的配置MediaClient。

SDK目录结构

```
baidubce
    ├─── auth              //公共权限目录
    ├─── services          //服务公共目录
    │    └─── media        //media目录
    └─── http              //Http请求模块
```

快速入门

1.初始化一个MediaClient。

MediaClient是与Media服务交互的客户端，所有Media操作都是通过MediaClient完成的。用户可以参考 配置MediaClient，完成初始化客户端的操作。

2.新建一个Pipeline(任务队列)。

通过Pipeline，用户可以更灵活地管理转码任务。当用户创建一个Job(任务)时，用户必须指定一个队列。

用户在创建队列时，需要为队列指定队列的名称、队列的类型（免费型或私有型）、一组源多媒体资源所属的Bucket与目标多媒体资源所属的的Bucket。输入和输出的Bucket可以是不同的。

3.查看Preset(多媒体模板)

查看系统预设的多媒体模板。模板是一个视频资源在做转码计算时所需参数的集合。用户可以更简便的将一个模板应用于一个和多个视频的转码任务，以使这些任务输出相同规格的目标多媒体资源。

音视频转码为用户预设了丰富且完备的系统模板，以满足用户对于目标规格在格式、码率、分辨率、加解密、水印等诸多方向

上的普遍需求，对于不希望过多了解音视频复杂技术背景的用户来说，是最佳的选择。

4.新建一个Job(任务)。

创建一个Job执行多媒体转码任务。每个任务将一个原始的多媒体资源转码成目标规格的多媒体资源。因此，任务和转码的目标是一一对应的，也就是说如果用户需要将一个原始视频规格转换成三种目标规格，比如从AVI格式转码成FLV/MP4/HLS格式，那么用户将会需要创建三个任务。

用户在创建任务时，需要为任务指定所属的队列、所需应用的转码模板以及原始音视频资源的BOS Key以及目标音视频资源BOS Key。

5.查询指定Object多媒体格式信息

用户通过BOS Bucket+Key获取指定多媒体文件的媒体信息。

MediaClient

配置MediaClient

MediaClient是Media服务的Python客户端，为调用者与Media服务进行交互提供了一系列的方法。

在新建MediaClient之前，需要先创建配置文件对MediaClient进行配置，以下将此配置文件命名为`conf.py`，具体配置信息如下所示：

```
 #!/usr/bin/env python
#coding=utf-8

#导入Python标准日志模块
import logging

#从Python SDK导入Media配置管理模块以及安全认证模块
from baidubce.bce_client_configuration import BceClientConfiguration
from baidubce.auth.bce_credentials import BceCredentials

#设置MediaClient的Host，Access Key ID和Secret Access Key
media_host = "http://media.bj.baidubce.com"
access_key_id = "your-access-key-id"
secret_access_key = "your-secret-access-key"

#设置日志文件的句柄和日志级别
logger = logging.getLogger('baidubce.services.media.mediaclient')
fh = logging.FileHandler("sample.log")
fh.setLevel(logging.DEBUG)

#设置日志文件输出的顺序、结构和内容
formatter = logging.Formatter('%(asctime)s - %(name)s - %(levelname)s - %(message)s')
fh.setFormatter(formatter)
logger.setLevel(logging.DEBUG)
logger.addHandler(fh)

#创建BceClientConfiguration
config = BceClientConfiguration(credentials=BceCredentials(access_key_id, secret_access_key), endpoint = media_host)
```

注意：

1.在上面的代码中，变量AK与SK是系统分配给用户的，用于标识用户，为访问Media做签名验证。其中AK对应控制台中的“Access Key ID”，SK对应控制台中的“Access Key Secret”，获取方式请参考《操作指南 管理ACCESSKEY》。

2.ENDPOINT参数只能用指定的包含Region的域名来进行定义，目前MCP提供了“华北-北京”、“华南-广州”和“华东-苏州”三个Region。详细的服务域名可以参考：服务域名。随着Region的增加将会开放其他可以支持的域名。

新建MediaClient

在完成上述配置之后，参考如下代码新建一个MediaClient。

```
 import conf
import sys
from baidubce import exception
from baidubce.services import media
from baidubce.services.media.media_client import MediaClient

# create new MediaClient
client = MediaClient(conf.config)
reload(sys)
sys.setdefaultencoding('utf-8')
print client.list_pipelines()
```

参数说明

Python SDK在`baidubce\bce_client_configuration.py`中默认设置了一些基本参数，若用户想要对参数的值进行修改，可以参考此文件创建自身的参数配置函数，并在构造MediaClient的时候传入，传入代码参考如下：

```
 from baidubce.retry_policy import BackOffRetryPolicy
from baidubce.bce_client_configuration import BceClientConfiguration
from baidubce.auth.bce_credentials import BceCredentials
from baidubce.protocol import HTTP
from baidubce.region import BEIJING

my_policy = BackOffRetryPolicy(max_error_retry = 3,
        max_delay_in_millis=20 * 1000,
        base_interval_in_millis=300)

my_config = BceClientConfiguration(
        credentials = BceCredentials('your-access-key-id', 'your-secret-access-key'),
        endpoint = media_host,
        protocol = baidubce.protocol.HTTP,
        region = baidubce.region.BEIJING,
        connection_timeout_in_mills = 50 * 1000,
        send_buf_size = 1024 * 1024,
        recv_buf_size = 10 * 1024 * 1024,
        retry_policy = my_policy)


# create MediaClient with my config
my_client = MediaClient(my_config)

pipelines = client.list_pipelines()
for pipeline in pipelines.pipelines:
  print pipeline
```

参数说明如下：

参数	说明	默认值
PROTOCOL	协议	baidubce.protocol.HTTP
REGION	区域	baidubce.region.BEIJING（目前只支持北京地区）
CONNECTION_TIMEOUT_IN_MILLIS	请求超时时间（单位：毫秒）	120 * 1000
SOCKET_TIMEOUT_IN_MILLIS	通过打开的连接传输数据的超时时间（单位：毫秒）	300 * 1000（0指的是无限等待，若设置非0数值需要对文件大小和网速进行评估，否则上传大文件时会产生超时）
SEND_BUF_SIZE	发送缓冲区大小	5 *1024* 1024
RECV_BUF_SIZE	接收缓冲区大小	5 *1024* 1024
retry_policy	重试逻辑	最大重试次数3次，超时时间为20 * 1000毫秒，重试间隔300毫秒

相关说明

MediaClient将可选的参数封装到config中，每一个方法具有的可选参数详见具体的接口使用方法介绍，现以create_pipeline方法为例，参考如下代码实现设置可选参数：

```
 #利用options在通过创建Pipeline传入指定可选参数
my_config = BceClientConfiguration(
        credentials = BceCredentials('your-access-key-id', 'your-secret-access-key'),
        endpoint = media_host,
        send_buf_size = 5 * 1024 * 1024)
client.create_pipeline('your_pipeline', 'your_source_bucket', 'your_source_bucket', config=my_config);
```

Pipeline（队列）

队列分为免费型与私有型：

- 免费型队列中的转码任务分享百度智能云为音视频转码所提供的约400路720P转码计算资源。
- 私有型队列需额外采购，以便更好的满足那些对于转码时效性和稳定性有更高要求的用户的业务需求。

用户可以利用队列实现任务优先级。用户通过创建多个队列达到区分任务优先级的目的，将大部分任务创建至普通优先级队列，将高优的任务放入高优先级的队列，以利用队列先到先服务的工作原理来实现任务的优先级调整。

> 队列接口的各参数含义和取值请参考Pipeline API。

新建Pipeline

如下代码可以新建一个Pipeline，默认的capacity值为20：

```
 pipeline_name = "your_pipeline";
source_bucket = "your_source_bucket";
target_bucket = "your_target_bucket";
# create a new pipeline
client.create_pipeline(pipeline_name, source_bucket, target_bucket);
```

列出全部Pipeline

如下代码可以列出用户所有的Pipeline：

```
response = client.list_pipelines()
for pipeline in response.pipelines:
  print pipeline
```

查询指定的Pipeline

若只是查询某个Pipeline，则使用如下代码：

```
pipeline_name = "your_pipeline"
response = client.get_pipeline(pipeline_name)
print response
```

删除Pipeline

如下代码可以删除一个Pipeline：

```
pipeline_name = "your_pipeline"
response = client.delete_pipeline(pipeline_name)
print response
```

需要注意的是，如果Pipeline有关联的Job未完成，则Pipeline无法被删除，必须等Job执行结束后才能成功删除。

更新指定的Pipeline

更新某个Pipeline的通知，使用如下代码：

```
pipeline_name = "your_pipeline"
notification = "your_notification"
pipeline = self.the_client.get_pipeline_for_update(pipeline_name)
pipeline.config.notification = notification
response = self.the_client.update_pipeline(pipeline_name, pipeline)
print response
```

更新pipeline的其他参数类似

Job（任务）

Job(任务)是音视频转码中最基本的执行单元，每个任务将一个原始的音视频资源转码成目标规格的音视频资源。因此，任务和转码的目标是一一对应的，也就是说如果用户需要将一个原始多媒体文件转换成三种目标规格，比如从AVI格式转码成FLV/MP4/HLS格式，那么用户将会需要创建三个任务。

> 任务接口的各参数含义和取值方法参考Job Transcoding API。

创建Job

用户在创建任务时，需要为任务指定所属的Pipeline、所需应用的Preset以及原始音视频资源的BOS Key以及目标音视频资源BOS Key。

如下代码创建一个Job, 并获取新创建的jobID：

```
pipeline_name = "your_pipeline"
source = {'sourceKey': 'your_source_key'}
target = {'targetKey': 'your_target_key', 'presetName': 'your_preset'}
response = client.create_job(pipeline_name, source, target)
print response
```

对Job进行条件筛选

创建任务后，您可以通过下述四个参数对缩略图任务和转码任务进行条件筛选，参数之间可相互配合：

- pipelineName: 队列名称
- job_status： 转码状态
- begin: createTime（创建时间）上限，用于筛选创建时间等于或者晚于begin的Job
- end: createTime（创建时间）下限，用于筛选创建时间等于或者早于end的Job

下述示例代码用于筛选指定队列下的所有Job:

```
 pipeline_name = "your_pipeline"
response = client.list_jobs(pipeline_name)
for job in response.jobs:
 print job
```

下述示例代码用于筛选指定队列下状态为“RUNNING”的所有Job:

```
 pipeline_name = "your_pipeline"
response = client.list_jobs(pipeline_name, job_status='RUNNING')
for job in response.jobs:
 print job
```

下述示例代码用于筛选指定队列下创建于2016年3月31日14:30:00之后的所有Job:

```
 pipeline_name = "your_pipeline"
response = client.list_jobs(pipeline_name, begin='2016-03-31T14:30:00Z')
for job in response.jobs:
 print job
```

下述示例代码用于筛选指定队列下创建于2016年3月31日14:30:00-15:00:00之间的所有Job:

```
 pipeline_name = "your_pipeline"
response = client.list_jobs(pipeline_name, begin='2016-03-31T14:30:00Z', end='2016-03-31T15:00:00Z')
for job in response.jobs:
 print job
```

查询指定的Job信息

用户可以通过如下代码通过jobId读取某个Job：

```
 job_id = "your_job"
response = client.get_job(job_id)
print response
```

Preset(模板)

模板是系统预设的对于一个视频资源在做转码计算时所需定义的集合。用户可以更简便的将一个模板应用于一个和多个视频的转码任务，以使这些任务输出相同规格的目标视频资源。

音视频转码为用户预设了丰富且完备的系统模板，以满足用户对于目标规格在格式、码率、分辨率、加解密、水印等诸多方向上的普遍需求，对于不希望过多了解音视频复杂技术背景的用户来说，是最佳的选择。百度为那些在音视频技术上有着丰富积累的用户，提供了可定制化的转码模板，以帮助他们满足复杂业务条件下的转码需求。

当用户仅需对于音视频的容器格式做变化时，百度提供Transmux模板帮助用户以秒级的延迟快速完成容器格式的转换，比如从MP4转换成HLS，而保持原音视频的属性不变。

> 模板接口的各参数含义和取值方法参考Preset-API。

查询当前用户Preset及所有系统Preset

用户可以通过如下代码查询所有的Preset：

```
response = client.list_presets()

for preset in response.presets:
  print preset
```

查询指定的Preset信息

用户可以通过如下代码指定的某个Preset：

```
preset_name = "your_preset"
response = client.get_preset(preset_name)
print response
```

创建Preset

如果系统预设的Preset无法满足用户的需求，用户可以自定义自己的Preset。根据不同的转码需求，可以使用不同的接口创建Preset。

创建仅支持容器格式转换的Preset

如下代码创建仅执行容器格式转换Preset：

```
preset_name = "your_preset"
container = "hls"
response = client.create_preset(preset_name, container)
print response
```

创建音频文件的转码Preset，不需要截取片段和加密

如果创建一个不需要截取片段和加密的音频文件转码Preset，可以参考如下代码：

```
preset_name = "your_preset"
container = "mp3"
audio = {'bitRateInBps': 25600, 'sampleRateInHz': 32000, 'channels': 1}
response = client.create_preset(preset_name, container, audio=audio)
print response
```

创建音频文件转码Preset，需要设置片段截取属性和加密属性

如果创建一个支持截取片段和加密的音频文件转码Preset，可以参考如下代码：

```
  preset_name = "your_preset"
container = "mp4"

clip = {'startTimeInSecond':0, 'durationInSecond': 50}
audio = {'bitRateInBps': 1980, 'sampleRateInHz': 32000, 'channels': 1}
encryption = {'aesKey': 'abcdefghij123456','strategy': 'Fixed'}
response = client.create_preset(preset_name, container, clip=clip, audio=audio, encryption=encryption)
print response
```

创建视频文件转码Preset，不需要截取片段和加密

如果创建一个不需要截取片段和加密的音频文件转码Preset，可以参考如下代码：

```
  preset_name = "your_preset"
container = "mp4"
codecOptions = {'profile': 'baseline'}
video = {
    'codec': 'h264',
    'codecOptions': codecOptions,
    'bitRateInBps': 32000,
    'maxFrameRate': 30,
    'maxWidthInPixel': 4096,
    'maxHeightInPixel': 96,
    'sizingPolicy': 'stretch'
}
audio = {'bitRateInBps': 1980, 'sampleRateInHz': 32000, 'channels': 1}
response = client.create_preset(preset_name, container, video=video, audio=audio)
print response
```

创建视频文件转码Preset，需要设置片段截取属性和加密属性

如果创建一个截取片段和加密的音频文件转码Preset，可以参考如下代码：

```
  preset_name = "your_preset"
container = "mp4"
codecOptions = {'profile': 'baseline'}
video = {
    'codec': 'h264',
    'codecOptions': codecOptions,
    'bitRateInBps': 32000,
    'maxFrameRate': 30,
    'maxWidthInPixel': 4096,
    'maxHeightInPixel': 96,
    'sizingPolicy': 'stretch'
}
clip = {'startTimeInSecond':0, 'durationInSecond': 50}
audio = {'bitRateInBps': 1980, 'sampleRateInHz': 32000, 'channels': 1}
encryption = {'aesKey': 'abcdefghij123456','strategy': 'Fixed'}
response = client.create_preset(preset_name, container, video=video, clip=clip, audio=audio, encryption=encryption)
print response
```

创建Preset，指定所有的参数

如果需要定制所有配置参数，可以参考如下代码：

```
 preset_name = "your_preset"

container = "mp4"
desc = 'My Desc'
transmux = True
codecOptions = {'profile': 'baseline'}
video = {
    'codec': 'h264',
    'codecOptions': codecOptions,
    'bitRateInBps': 32000,
    'maxFrameRate': 30,
    'maxWidthInPixel': 4096,
    'maxHeightInPixel': 96,
    'sizingPolicy': 'stretch'
}
clip = {'startTimeInSecond':0, 'durationInSecond': 50}
audio = {'bitRateInBps': 1980, 'sampleRateInHz': 32000, 'channels': 1}
encryption = {'aesKey': 'abcdefghij123456','strategy': 'Fixed'}
response = client.create_preset(preset_name, container, transmux=transmux, description=desc, video=video, clip=clip,
audio=audio, encryption=encryption)
print response
```

更新Preset

用户可以修改自己创建的模板。

如下代码可以更新指定模板的音频目标码率：

```
 preset_name = "your_preset"
preset = self.the_client.get_preset_for_update(preset_name)
preset.audio.bitRateInBps = 256000
response = self.the_client.update_preset(preset_name, preset)
print response
```

更新其他参数类似

Mediainfo(媒体信息)

对于BOS中某个Object，可以通过下面代码获其媒体信息：

```
 bucket = "your_bucket"
key = "your_key"
response = client.get_mediainfo_of_file(bucket, key)
print response
```

Thumbnail-Job(缩略图任务)

创建缩略图任务

如果要创建一个缩略图任务，可以参考如下代码：

```
 pipeline_name = 'a3'
source = {'key': '36A2014.mp4'}
response = client.create_thumbnail_job(pipeline_name,  source)
print response
```

接口返回的是包含了jobId的一个对象。

查询指定缩略图任务

如果需要获取一个已创建的缩略图任务的信息，可以参考如下代码：

```
job_id = "your jobId"
response = client.get_thumbnail_job(job_id)
print response
```

查询指定队列的缩略图任务信息

如果需要获取一个队列里的全部缩略图任务的信息，可以参考如下代码：

```
pipeline_name = "your pipeline"
response = client.list_thumbnail_jobs_by_pipeline(pipeline_name)
print response
```

Watermark(水印)

创建水印

如果需要创建一个水印，可以参考如下代码：

```
bucket = "your_bucket"
key = "your_key"
response = client.create_watermark(bucket, key)
print response
```

接口返回的是包含了watermarkId的一个对象。

查询指定水印

如果需要查询已创建的水印，可以参考如下代码：

```
watermark_id = "your_watermark_id"
response = client.get_watermark(watermark_id)
print response
```

查询当前用户水印

如果需要查询出本用户所创建的全部水印，可以参考如下代码：

```
response = client.list_watermarks()
print response
```

删除水印

如果需要删除某个已知watermarkId的水印，可以参考如下代码：

```
watermark_id = "your watermark id"
response = client.delete_watermark(watermark_id)
print response
```

Notification(通知)

创建通知

如果需要创建一个通知，可以参考如下代码：

```
  name = "your_notification"
endpoint = "your_endpoint"
response = client.create_notification(name, endpoint)
print response
```

查询指定通知

如果需要查询一个通知，可以参考如下代码：

```
  name = "your_notification"
response = client.get_notification(name)
print response
```

查询当前用户通知

如果需要查询出本用户所创建的全部通知，可以参考如下代码：

```
  response = client.list_notifications()
print response
```

删除通知

如果需要删除某个已知name的通知，可以参考如下代码：

```
  name = "your_notification"
response = client.delete_notification(name)
print response
```

异常处理

系统异常

Media异常提示有如下三种方式：

异常方法	说明
BceHttpClientError	重试时抛出的异常
--last_error	最后一次重试时抛出的异常
----BceClientError	Media客户端产生的异常
----BceInvalidArgumentError	传递参数产生的异常
----BceServerError	Media服务器产生的异常

用户可以使用try获取某个事件所产生的异常：

```
  from baidubce.exception import BceHttpClientError
from baidubce.exception import BceServerError
from baidubce.exception import BceClientError
try:
    watermark_id = "non_exist"
    client.delete_watermark(watermark_id)
except BceHttpClientError as e:
    print "Cannot delete the watermark: ", e.message
```

返回为：

```
 Cannot delete the watermark:  Unable to execute HTTP request. Retried 0 times. A
ll trace backs:
>>>>Traceback (most recent call last):
>>>>  File "C:\tools\Python27\lib\site-packages\baidubce\http\bce_http_client.py
", line 183, in send_request
>>>>    if handler_function(http_response, response):
>>>>  File "C:\tools\Python27\lib\site-packages\baidubce\http\handler.py", line
71, in parse_error
>>>>    raise bse
>>>>BceServerError: watermark: non_exist does not exist

也可以用这种方式直接获取原始错误信息：
print "Cannot delete the watermark: ", e.last_error.message
得到：
Cannot delete the watermark: watermark: non_exist does not exist
```

参数异常

Media Python SDK的每个调用都有一些类型固定不可以为空的参数，若该参数传入为空值则返回BceClientError，若该参数传入类型错误则返回TypeError。

版本变更记录

0.8.10

在支持pipelineName（队列名称）筛选参数的基础上，进一步增加下述参数方便用户对缩略图任务和转码任务进行条件筛选：

- job_status： 转码状态
- begin: createTime（创建时间）上限，用于筛选创建时间等于或者晚于begin的Job
- end: createTime（创建时间）下限，用于筛选创建时间等于或者早于end的Job

0.8.35

在创建Preset（模板）时，支持新的可选请求参数：

- trans_config： 转码配置信息，可以配置transMode：String（转码模式）
- extra_config：转码额外配置，可以配置watermarkDisableWhitelist：String（不加水印的条件）、segmentDurationInSecond：Number（分片时长）等参数。详见API文档

0.8.36

- 新增通知相关接口
- 新增更新pipeline、preset接口

PHP-SDK

安装MCT-PHP-SDK

安装SDK包

1. 从官方网站下载PHP SDK压缩包。
2. 解压安装包并浏览SDK目录：

```
BaiduBce.phar
├──src
│    └── BaiduBce
│        ├── Auth              //BCE签名相关
│        ├── Exception          //BCE客户端的异常
│        ├── Http              //BCE的Http通信相关
│        ├── Log               //BCE日志
│        ├── Services
│        │    └── Media                //Media主目录，此目录必须保留
│        │        └── MediaClient.php    //Media操作类，所有操作可以通过MediaClient类可以完成
│        └── Util              //BCE公用工具
└──vendor                      //第三方库
```

3. 在脚本文件中添加以下代码并保存：

```
include 'BaiduBce.phar';
require 'YourConf.php';
```

有关配置文件的引用，请参考配置MediaClient。

安装MediaClient

配置MediaClient

MediaClient是MCT服务的PHP客户端，为开发者与MCT服务进行交互提供了一系列的方法。

在新建MediaClient之前，需要先创建配置文件对MediaClient进行配置，以下将此配置文件命名为YourConf.php，具体配置信息如下所示：

```
// 报告所有 PHP 错误
error_reporting(-1);

define('__MEDIA_CLIENT_ROOT', dirname(__DIR__));

// 设置MediaClient的Access Key ID、Secret Access Key和ENDPOINT
$MEDIA_TEST_CONFIG =
    array(
        'credentials' => array(
            'ak' => 'your-access-key-id',
            'sk' => 'your-secret-access-key',
        ),
        'endpoint' => 'http://media.bj.baidubce.com',
    );

// 设置log的格式和级别
$__handler = new \Monolog\Handler\StreamHandler(STDERR, \Monolog\Logger::DEBUG);
$__handler->setFormatter(
    new \Monolog\Formatter\LineFormatter(null, null, false, true)
);
\BaiduBce\Log\LogFactory::setInstance(
    new \BaiduBce\Log\MonoLogFactory(array($__handler))
);
\BaiduBce\Log\LogFactory::setLogLevel(\Psr\Log\LogLevel::DEBUG);
```

注意：

1.在上面的代码中，变量AK与SK是系统分配给用户的，用于标识用户，为访问Media做签名验证。其中AK对应控制台中的“Access Key ID”，SK对应控制台中的“Access Key Secret”，获取方式请参考《操作指南 管理ACCESSKEY》。

2.ENDPOINT参数只能用指定的包含Region的域名来进行定义，目前MCP提供了“华北-北京”、“华南-广州”和“华东-苏州”三个Region。详细的服务域名可以参考：服务域名。随着Region的增加将会开放其他可以支持的域名。

新建MediaClient

在完成上述配置之后，参考如下代码新建一个MediaClient。

```
 //使用PHP SDK，并且使用自定义配置文件
include 'BaiduBce.phar';
require 'YourConf.php';

use BaiduBce\Services\Media\MediaClient;

//调用配置文件中的参数
global $MEDIA_TEST_CONFIG;
//新建MediaClient
$client = new MediaClient($MEDIA_TEST_CONFIG);
```

参数说明

PHP SDK在`\BaiduBce\Bce.php`中默认设置了一些基本参数，若用户想要对参数的值进行修改，可以参考此文件创建自身的参数配置函数，并在构造MediaClient的时候传入，传入代码参考如下：

```
public function CustomizedConfig() {
    $customizedConfig = array(
        BceClientConfigOptions::PROTOCOL => 'http',
        BceClientConfigOptions::REGION => 'bj',
        BceClientConfigOptions::CONNECTION_TIMEOUT_IN_MILLIS => 120 * 1000,
        BceClientConfigOptions::SOCKET_TIMEOUT_IN_MILLIS => 300 * 1000,
        BceClientConfigOptions::SEND_BUF_SIZE => 5 * 1024 * 1024,
        BceClientConfigOptions::RECV_BUF_SIZE => 5 * 1024 * 1024,
        BceClientConfigOptions::CREDENTIALS => array(
            'ak' => 'your-access-key-id',
            'sk' => 'your-secret-access-key',
        ),
        'endpoint' => 'your-endpoint',
    );

    //利用自定义配置创建MediaClient
    $customizedClient = new MediaClient($customizedConfig);

    //通过自定义配置调用方法
    $options = array('config'=>$customizedConfig);
    $this->client->listPipelines($options);
}
```

参数说明如下：

参数	说明	默认值
PROTOCOL	协议	http
REGION	区域	bj（目前只支持北京地区）
CONNECTION_TIMEOUT_IN_MILLIS	请求超时时间（单位：毫秒）	50 * 1000
SOCKET_TIMEOUT_IN_MILLIS	通过打开的连接传输数据的超时时间（单位：毫秒）	0（指的是无限等待，若设置非0数值需要对文件大小和网速进行评估，否则上传大文件时会产生超时）
SEND_BUF_SIZE	发送缓冲区大小	1024 * 1024
RECV_BUF_SIZE	接收缓冲区大小	10 ** *1024* 1024

相关说明

MediaClient将可选的参数封装到`$options`中，每一个方法具有的可选参数详见具体的接口使用方法介绍，现以`createPipeline`方法为例，参考如下代码实现设置可选参数：

```
 //利用options在通过创建Pipeline传入指定可选参数
$options = array(
    'description' => 'This is a test pipeline',
    'pipelineConfig' => array(
        'capacity' => 15,
    ),
);
$client->createPipeline($pipelineName, $sourceBucket, $targetBucket, $options);
```

注意：不要把`null`传入`$options`中，否则调用时会抛出异常。

快速入门

1. 初始化一个MediaClient。

   MediaClient是与MCP服务交互的客户端，所有Media操作都是通过MediaClient完成的。用户可以参考新建MediaClient，完成初始化客户端的操作。

2. 新建一个Pipeline(任务队列)。

   通过Pipeline，用户可以更灵活地管理转码任务。当用户创建一个Job(任务)时，用户必须指定一个队列。

   用户在创建队列时，需要为队列指定队列的名称、队列的类型（免费型或独享型）、一组源多媒体资源所属的Bucket与目标多媒体资源所属的的Bucket。输入和输出的Bucket可以是不同的。

3. 查看Preset(多媒体模板)

   查看系统预设的多媒体模板。模板是一个视频资源在做转码计算时所需参数的集合。用户可以更简便的将一个模板应用于一个和多个视频的转码任务，以使这些任务输出相同规格的目标多媒体资源。

   音视频转码为用户预设了丰富且完备的系统模板，以满足用户对于目标规格在格式、码率、分辨率、加解密、水印等诸多方向上的普遍需求，对于不希望过多了解音视频复杂技术背景的用户来说，是最佳的选择。

4. 新建一个Job(任务)。

   创建一个Job执行多媒体转码任务。每个任务将一个原始的多媒体资源转码成目标规格的多媒体资源。因此，任务和转码的目标是一一对应的，也就是说如果用户需要将一个原始视频规格转换成三种目标规格，比如从AVI格式转码成FLV/MP4/HLS格式，那么用户将会需要创建三个任务。

   用户在创建任务时，需要为任务指定所属的队列、所需应用的转码模板以及原始音视频资源的BOS Key以及目标音视频资源BOS Key。

5. 查询指定Object多媒体格式信息

用户通过BOS Bucket+Key获取指定多媒体文件的媒体信息。

开发者指南

Pipeline（队列）

队列分为免费型与专享性：

- 免费型队列中的转码任务分享百度智能云为音视频转码所提供的约400路720P转码计算资源。
- 专享型队列需额外采购，以便更好的满足那些对于转码时效性和稳定性有更高要求的用户的业务需求。

用户可以利用队列实现任务优先级。用户通过创建多个队列达到区分任务优先级的目的，将大部分任务创建至普通优先级队列，将高优的任务放入高优先级的队列，以利用队列先到先服务的工作原理来实现任务的优先级调整。

新建Pipeline

如下代码可以新建一个Pipeline，默认的capacity值为20：

```
 $pipelineName = "your_pipeline";
$sourceBucket = "your_source_bucket";
$targetBucket = "your_source_bucket";
// 新建一个Pipeline
$client->createPipeline($pipelineName, $sourceBucket, $targetBucket);
```

列出全部Pipeline

如下代码可以列出用户所有的Pipeline：

```
 / 获取用户的Pipeline列表
$response = $client->listPipelines();
// 遍历Pipeline列表
foreach ($response->pipelines as $pipeline) {
    print json_encode($pipeline);
}
```

查询指定的Pipeline

若只是查询某个Pipeline，则使用如下代码：

```
 $pipelineName = "your_pipeline";
$response = $client->getPipeline($pipelineName);
print json_encode($response);
```

修改Pipeline

如下代码可以修改一个Pipeline的通知：

```
 $pipelineName = "your_pipeline";
$pipeline = $this->client->getPipeline($pipelineName);
$pipeline->config->notification = "your_notification";
$this->client->updatePipeline($pipelineName, json_decode(json_encode( $pipeline),true));
```

修改其他参数类似

删除Pipeline

如下代码可以删除一个Pipeline：

```
$pipelineName = "your_pipeline";
client->deletePipeline(pipelineName);
```

需要注意的是，如果Pipeline有关联的Job未完成，则Pipeline无法被删除，必须等Job执行结束后才能成功删除。

Job（任务）

Job(任务)是音视频转码中最基本的执行单元，每个任务将一个原始的音视频资源转码成目标规格的音视频资源。因此，任务和转码的目标是一一对应的，也就是说如果用户需要将一个原始多媒体文件转换成三种目标规格，比如从AVI格式转码成FLV/MP4/HLS格式，那么用户将会需要创建三个任务。

创建Job

用户在创建任务时，需要为任务指定所属的Pipeline、所需应用的Preset以及原始音视频资源的BOS Key以及目标音视频资源BOS Key。

如下代码创建一个Job, 并获取新创建的jobID：

```
$pipelineName = "your_pipeline";
$sourceKey = "your_source_key";
$targetKey = "your_target_key";
$presetName = "your_preset";
$client->createSimpleJob($pipelineName, $sourceKey, $targetKey, $presetName);
```

对Job进行条件筛选

创建任务后，您可以通过下述四个参数对转码任务进行条件筛选，参数之间可相互配合：

- pipelineName: 队列名称
- jobStatus： 转码状态
- begin: createTime（创建时间）上限，用于筛选创建时间等于或者晚于begin的Job
- end: createTime（创建时间）下限，用于筛选创建时间等于或者早于end的Job

下述示例代码用于筛选指定队列下的所有Job:

```
$pipelineName = "your_pipeline";
$response = $client->listJobs($pipelineName);
foreach($response->jobs as $job) {
    print json_encode($job);
}
```

下述示例代码用于筛选指定队列下状态为“RUNNING”的所有Job:

```
$pipelineName = "your_pipeline";
$response = $client->listJobs($pipelineName, "RUNNING");
foreach($response->jobs as $job) {
    print json_encode($job);
}
```

下述示例代码用于筛选指定队列下创建于2016年3月31日14:30:00之后的所有Job:

```
$pipelineName = "your_pipeline";
$response = $client->listJobs($pipelineName, null, "2016-03-31T14:30:00Z");
foreach($response->jobs as $job) {
    print json_encode($job);
}
```

下述示例代码用于筛选指定队列下创建于2016年3月31日14:30:00-15:00:00之间的所有Job:

```
$pipelineName = "your_pipeline";
$response = $client->listJobs($pipelineName, null, "2016-03-31T14:30:00Z", "2016-03-31T15:00:00Z");
foreach($response->jobs as $job) {
    print json_encode($job);
}
```

查询指定的Job信息

用户可以通过如下代码通过jobId读取某个Job：

```
$jobId = "your_job";
$response = $client->getJob($jobId);
print json_encode($response);
```

Preset(模板)

模板是系统预设的对于一个视频资源在做转码计算时所需定义的集合。用户可以更简便的将一个模板应用于一个和多个视频的转码任务，以使这些任务输出相同规格的目标视频资源。

音视频转码为用户预设了丰富且完备的系统模板，以满足用户对于目标规格在格式、码率、分辨率、加解密、水印等诸多方向上的普遍需求，对于不希望过多了解音视频复杂技术背景的用户来说，是最佳的选择。百度为那些在音视频技术上有着丰富积累的用户，提供了可定制化的转码模板，以帮助他们满足复杂业务条件下的转码需求。

当用户仅需对于音视频的容器格式做变化时，百度提供Transmux模板帮助用户以秒级的延迟快速完成容器格式的转换，比如从MP4转换成HLS，而保持原音视频的属性不变。

查询当前用户Preset及所有系统Preset

用户可以通过如下代码查询所有的Preset：

```
$response = $client->listPresets();
foreach($response->presets as $preset) {
    print json_encode($preset);
}
```

查询指定的Preset信息

用户可以通过如下代码指定的某个Preset：

```
$presetName = "your_preset";
$response = $client->getPreset($presetName);
print json_encode($response);
```

创建Preset

如果系统预设的Preset无法满足用户的需求，用户可以自定义自己的Preset。根据不同的转码需求，可以使用不同的接口创建Preset。

创建仅支持容器格式转换的Preset

如下代码创建仅执行容器格式转换Preset：

```
 $presetName = "your_preset";

//指定转换后的容器格式，可选值：mp4, flv, hls, mp3, m4a
$container = "mp4";

//指定仅需要容器格式转换
$options = array(
    'transmux' => true,
);

$client->createPreset($presetName, $container, $options);
```

创建音频文件的转码Preset，不需要截取片段和加密

如果创建一个不需要截取片段和加密的音频文件转码Preset，可以参考如下代码：

```
 $presetName = "your_preset";
$container = "mp4";
$options = array(
    "audio" => array(
        "bitRateInBps" => 80000,
        "sampleRateInHz" => 22050,
        "channels" => 2 ,
    ),
);
$client->createPreset($presetName, $container, $options);
```

创建音频文件转码Preset，需要设置片段截取属性和加密属性

如果创建一个支持截取片段和加密的音频文件转码Preset，可以参考如下代码：

```
 $presetName = "your_preset";
$container = "mp4";
$options = array(
    "audio" => array(
        "bitRateInBps" => 80000,
        "sampleRateInHz" => 22050,
        "channels" => 2 ,
    ),
    "clip" => array(
        "startTimeInSecond" => 0,
        "durationInSecond" => 60,
    ),
    "encryption" => array(
        "aesKey" => "abcdefghij1234567",
        "strategy" => "Fixed",
    ),
);

$client->createPreset($presetName, $container, $options);
```

创建视频文件转码Preset，不需要截取片段和加密

如果创建一个不需要截取片段和加密的音频文件转码Preset，可以参考如下代码：

```
 $presetName = "your_preset";
$container = "mp4";
$options = array(
    "audio" => array(
        "bitRateInBps" => 80000,
        "sampleRateInHz" => 22050,
        "channels" => 2 ,
    ),
    "video" => array(
        "codec" => "h264",
        "codecOptions" =>array(
            "profile" => "baseline"
        ),
        "bitRateInBps" => 32000,
        "maxFrameRate" => 30,
        "maxWidthInPixel" => 4096,
        "maxHeightInPixel"=> 96,
        "sizingPolicy" => "stretch",
    ),
);

$client->createPreset($presetName, $container, $options);
```

创建视频文件转码Preset，需要设置片段截取属性和加密属性

如果创建一个截取片段和加密的音频文件转码Preset，可以参考如下代码：

```
 $presetName = "your_preset";
$container = "mp4";
$options = array(
    "audio" => array(
        "bitRateInBps" => 80000,
        "sampleRateInHz" => 22050,
        "channels" => 2 ,
    ),
    "video" => array(
        "codec" => "h264",
        "codecOptions" =>array(
            "profile" => "baseline"
        ),
        "bitRateInBps" => 32000,
        "maxFrameRate" => 30,
        "maxWidthInPixel" => 4096,
        "maxHeightInPixel"=> 96,
        "sizingPolicy" => "stretch",
    ),
    "clip" => array(
        "startTimeInSecond" => 0,
        "durationInSecond" => 60,
    ),
    "encryption" => array(
        "aesKey" => "abcdefghij1234567",
        "strategy" => "Fixed",
    ),
);

$client->createPreset($presetName, $container, $options);
```

创建Preset，指定所有的参数

如果需要定制所有配置参数，可以参考如下代码：

```
 $presetName = "your_preset";
$container = "mp4";
$options = array(
    "description" => "this is a demo preset",
    "transmux" => "false",
    "audio" => array(
        "bitRateInBps" => 80000,
        "sampleRateInHz" => 22050,
        "channels" => 2 ,
    ),
    "video" => array(
        "codec" => "h264",
        "codecOptions" =>array(
            "profile" => "baseline"
        ),
        "bitRateInBps" => 32000,
        "maxFrameRate" => 30,
        "maxWidthInPixel" => 4096,
        "maxHeightInPixel"=> 96,
        "sizingPolicy" => "stretch",
    ),
    "clip" => array(
        "startTimeInSecond" => 0,
        "durationInSecond" => 60,
    ),
    "encryption" => array(
        "aesKey" => "abcdefghij1234567",
        "strategy" => "Fixed",
    ),
);

$client->createPreset($presetName, $container, $options);
```

修改Preset

如下代码可以更新一个Preset：

```
 $presetName = "your_preset";
$preset = $this->client->getPreset($presetName);
$preset->audio->bitRateInBps = 800000;
$this->client->updatePreset($presetName, json_decode(json_encode( $preset),true));
```

修改其他参数类似

Mediainfo(媒体信息)

对于BOS中某个Object，可以通过下面代码获其媒体信息：

```
 $bucket = "your_bucket";
$key = "your_key";
$response = $client->getMediaInfoOfFile($bucket, $key);
print json_encode($response);
```

Thumbnail Job(缩略图任务)

创建缩略图任务

如果要创建一个缩略图任务，可以参考如下代码：

```
 $pipelineName = "your_pipeline";
$source = array("key" => "your_key");

//设置可选参数，包括缩略图输出方法及截取规则
$options = array(
    "target" => array(
        "keyPrefix" => "thumbnail_target_key_prefix",
        "format" => "jpg",
        "sizingPolicy" => "keep",
        "widthInPixel" => 600,
        "heightInPixel" => 400,
    ),
    "capture" => array(
        "mode" => "manual",
        "startTimeInSecond" => 0,
        "endTimeInSecond" => 50,
        "intervalInSecond" => 10,
    ),
);

$response = $client->createThumbnailJob($pipelineName, $source, $options);
print $response->jobId;
```

接口返回的是包含了jobId的一个对象。

查询指定缩略图任务

如果需要获取一个已创建的缩略图任务的信息，可以参考如下代码：

```
 $jobId = "your_jobId";
$response = client->getThumbnailJob($jobId);
print json_encode($response);
```

对Thumbnail Job进行条件筛选

创建缩略图任务后，您可以通过下述四个参数对缩略图任务进行条件筛选，参数之间可相互配合：

- pipelineName: 队列名称
- jobStatus： 转码状态
- begin: createTime（创建时间）上限，用于筛选创建时间等于或者晚于begin的Thumbnail Job
- end: createTime（创建时间）下限，用于筛选创建时间等于或者早于end的Thumbnail Job

下述示例代码用于筛选指定队列下的所有Thumbnail Job:

```
 $pipelineName = "your_pipeline";
$response = $client->listThumbnailJobsByPipelineName($pipelineName);
print json_encode($response);
```

下述示例代码用于筛选指定队列下状态为“RUNNING”的所有Thumbnail Job:

```
 $pipelineName = "your_pipeline";
$response = $client->listThumbnailJobsByPipelineName($pipelineName, "RUNNING");
print json_encode($response);
```

下述示例代码用于筛选指定队列下创建于2016年3月31日14:30:00之后的所有Thumbnail Job:

```
$pipelineName = "your_pipeline";
$response = $client->listThumbnailJobsByPipelineName($pipelineName, "2016-03-31T14:30:00Z");
print json_encode($response);
```

下述示例代码用于筛选指定队列下创建于2016年3月31日14:30:00-15:00:00之间的所有Thumbnail Job:

```
$pipelineName = "your_pipeline";
$response = $client->listThumbnailJobsByPipelineName($pipelineName, "2016-03-31T14:30:00Z", "2016-03-31T15:00:00Z");
print json_encode($response);
```

Watermark(水印)

创建水印

可以使用BOS中的图片对象创建一个水印，后续可以在创建自定义转码模板时设置视频水印。

创建水印，使用默认位置参数

如果需要创建一个水印（水印的默认位置是左上），可以参考如下代码：

```
$bucket = "your_bucket";
$key = "your_key";

$response = $client->createWatermark(
    $bucket,
    $key
);
print $response->watermarkId;
```

接口返回的是包含了watermarkId的一个对象。

创建水印，指定位置参数

如果需要定制水印的位置参数，可以参考如下代码：

```
$bucket = "your_bucket";
$key = "your_key";
$options = array(
    //可选值：top/center/bottom
    "verticalAlignment" => "top",
    //可选值：left/center/right
    "horizontalAlignment" => "left",
    "verticalOffsetInPixel" => 0,
    "horizontalOffsetInPixel" => 0,
);

$client->createWatermark(
    $bucket,
    $key,
    $options
);
```

查询指定水印

如果需要查询已创建的水印，可以参考如下代码：

```
$watermarkId = "your_watermarkId";
$response = $client->getWatermark($watermarkId);
print json_encode($response);
```

查询当前用户水印

如果需要查询出本用户所创建的全部水印，可以参考如下代码：

```
$response = $client->listWatermarks();
print json_encode($response);
```

删除水印

如果需要删除某个已知watermarkId的水印，可以参考如下代码：

```
$watermarkId = "your_watermarkId";
$client->deleteWatermark($watermarkId);
```

Notification（通知）

通知功能可以在音视频转码任务状态转换时主动向开发者服务器推送消息。

新建通知

如下代码可以新建一个Notification：

```
$name= "your_notification";
$endpoint= "your_endpoint";
$this->client->createNotification($name, $endpoint)
);
```

列出全部Notification

如下代码可以列出用户所有的Notification：

```
// 获取用户的Notification列表
$response = $client->listNotifications();
// 遍历Notification列表
foreach ($response->notifications as $notification) {
    print json_encode($notification);
}
```

查询指定的Notification

若只是查询某个Notification，则使用如下代码：

```
$name = "your_notification";
$response = $client->getNotification($name);
print json_encode($response);
```

删除Notification

如下代码可以删除一个Notification：

```
$name = "your_notification";
client->deleteNotification($name);
```

异常处理

Media异常提示有如下三种方式：

异常方法	说明
BceBaseException	异常总集
BceClientException	客户端异常
BceServerException	服务器异常
InvalidArgumentException	系统自带异常，参数错误

用户可以使用try-catch获取某个事件所产生的异常：

```
 try {
    $client->listPresets();
}catch (\BaiduBce\Exception\BceBaseException $e) {
    print $e->getMessage();
    if (stcmp(gettype($e), "BceClientException") == 0) {
        print "Catch a client exception";
    }
    if (stcmp(gettype($e), "BceServerException") == 0) {
        print "Catch a server exception";
    }
}
```

版本变更记录

0.8.11

在支持pipelineName（队列名称）筛选参数的基础上，进一步增加下述参数方便用户对缩略图任务和转码任务进行条件筛选：

- jobStatus： 转码状态
- begin: createTime（创建时间）上限，用于筛选创建时间等于或者晚于begin的(Thumbnail) Job
- end: createTime（创建时间）下限，用于筛选创建时间等于或者早于end的(Thumbnail) Job

0.9.8

- 新增通知相关接口
- 新增更新pipeline、preset接口

Golang-SDK

简介

本文档主要介绍MCP GO SDK的使用。在使用本文档前，您需要先了解MCP的一些基本知识，并已开通了MCP服务。若您还不了解MCP，可以参考产品描述和入门指南。

安装Media-Go-SDK

运行环境

GO SDK可以在go1.3及以上环境下运行。

安装SDK

直接从github下载

使用go get工具从github进行下载：

```
go get github.com/baidubce/bce-sdk-go
```

SDK目录结构

```
 bce-sdk-go
|--auth                 //BCE签名和权限认证
|--bce                  //BCE公用基础组件
|--http                 //BCE的http通信模块
|--services              //BCE相关服务目录
|  |--media              //音视频处理MCP
|--util                 //BCE公用的工具实现
```

卸载SDK

预期卸载SDK时，删除下载的源码即可。

快速入门

1.初始化一个MediaClient。

MediaClient是与Media服务交互的客户端，所有Media操作都是通过MediaClient完成的。用户可以参考新建MediaClient，完成初始化客户端的操作。

2.新建一个Pipeline(任务队列)。

通过Pipeline，用户可以更灵活地管理转码任务。当用户创建一个Job(任务)时，用户必须指定一个队列。

用户在创建队列时，需要为队列指定队列的名称、队列的类型（免费型或独享型）、一组源多媒体资源所属的Bucket与目标多媒体资源所属的的Bucket。输入和输出的Bucket可以是不同的。

3.查看Preset(多媒体模板)

查看系统预设的多媒体模板。模板是一个视频资源在做转码计算时所需参数的集合。用户可以更简便的将一个模板应用于一个和多个视频的转码任务，以使这些任务输出相同规格的目标多媒体资源。

音视频转码为用户预设了丰富且完备的系统模板，以满足用户对于目标规格在格式、码率、分辨率、加解密、水印等诸多方向上的普遍需求，对于不希望过多了解音视频复杂技术背景的用户来说，是最佳的选择。

4.新建一个Job(任务)。

创建一个Job执行多媒体转码任务。每个任务将一个原始的多媒体资源转码成目标规格的多媒体资源。因此，任务和转码的目标是一一对应的，也就是说如果用户需要将一个原始视频规格转换成三种目标规格，比如从AVI格式转码成FLV/MP4/HLS格式，那么用户将会需要创建三个任务。

用户在创建任务时，需要为任务指定所属的队列、所需应用的转码模板以及原始音视频资源的BOS Key以及目标音视频资源BOS Key。

5.查询指定Object多媒体格式信息

用户通过BOS Bucket+Key获取指定多媒体文件的媒体信息。

MediaClient

初始化

确认Endpoint

在确认您使用SDK时配置的Endpoint时，可先阅读开发人员指南中关于使用须知的部分，理解Endpoint相关的概念。百度智能云目前开放了多区域支持，请参考区域选择说明。

目前支持“华北-北京”、“华南-广州”和“华东-苏州”三个区域。北京区域：`http://media.bj.baidubce.com`，广州区域：`http://media.gz.baidubce.com`，苏州区域：`http://media.su.baidubce.com`。对应信息为：

访问区域	对应Endpoint
BJ	media.bj.baidubce.com
GZ	media.gz.baidubce.com
SU	media.su.baidubce.com

获取密钥

要使用MCP，您需要拥有一个有效的AK(Access Key ID)和SK(Secret Access Key)用来进行签名认证。AK/SK是由系统分配给用户的，均为字符串，用于标识用户，为访问MCP做签名验证。

可以通过如下步骤获得并了解您的AK/SK信息：

注册账号

创建AK/SK

新建MCP Client

MCP Client是MCP服务的客户端，为开发者与MCP服务进行交互提供了一系列的方法。

使用AK/SK新建MCP Client

通过AK/SK方式访问MCP，用户可以参考如下代码新建一个MCP Client：

```
 import (
"github.com/baidubce/bce-sdk-go/services/media"
)

func main() {
 // 用户的Access Key ID和Secret Access Key
 ACCESS_KEY_ID, SECRET_ACCESS_KEY := <your-access-key-id>, <your-secret-access-key>

 // 用户指定的Endpoint
 ENDPOINT := <domain-name>

 // 初始化一个MCPClient
 MEDIA_CLIENT, err := media.NewClient(AK, SK, ENDPOINT)
}
```

在上面代码中，`ACCESS_KEY_ID`对应控制台中的“Access Key ID”，`SECRET_ACCESS_KEY`对应控制台中的“Access Key Secret”，获取方式请参考《相关参考 如何获取AKSK》。第三个参数`ENDPOINT`支持用户自己指定域名，如果设置为空字符串，会使用默认域名作为MCP的服务地址。

> **注意：**`ENDPOINT`参数需要用指定区域的域名来进行定义，如服务所在区域为北京，则为`http://media.bj.baidubce.com`。

Pipeline队列

队列分为免费型与专享型：

- 免费型队列中的转码任务分享百度智能云为音视频转码所提供的约400路720P转码计算资源。

- 专享型队列需额外采购，以便更好的满足那些对于转码时效性和稳定性有更高要求的用户的业务需求。

用户可以利用队列实现任务优先级。用户通过创建多个队列达到区分任务优先级的目的，将大部分任务创建至普通优先级队列，将高优的任务放入高优先级的队列，以利用队列先到先服务的工作原理来实现任务的优先级调整。

新建Pipeline

如下代码可以新建一个Pipeline。

```
 pipelineName := "test"
sourceBucket := "testBucket"
targetBucket := "targetBucket"
capacity := 10
err := MEDIA_CLIENT.CreatePipeline(pipelineName, sourceBucket, targetBucket, capacity)
if err != nil {
    fmt.Printf("create Pipeline error: %+v\n", err)
    return
}
fmt.Println("create pipeline success")
```

列出全部pipeline

如下代码可以列出用户所有的pipeline

```
 pipelines, err := MEDIA_CLIENT.ListPipelines()
if err != nil {
    fmt.Printf("list Pipeline error: %+v\n", err)
    return
}
fmt.Println("list pipeline success\n")
for _, pipeline := range pipelines.Pipelines {
 fmt.Printf("pipeline: %+v\n", pipeline)
}
```

查询指定pipeline

如下代码可以按照pipelineName查询pipeline。

```
 pipelineName := "test"
pipeline, err := MEDIA_CLIENT.GetPipeline(pipelineName)
if err != nil {
    fmt.Printf("list Pipeline error: %+v\n", err)
    return
}
fmt.Println("get pipeline success")
fmt.Printf("pipeline: %+v\n", pipeline)
```

删除pipeline

如下代码可以按照pipelineName删除pipeline。

```
 pipelineName := "test"
err := MEDIA_CLIENT.DeletePipeline(pipelineName)
if err != nil {
    fmt.Printf("delete Pipeline error: %+v\n", err)
    return
}
fmt.Println("delete pipeline success")
```

需要注意的是，如果Pipeline有关联的Job未完成，则Pipeline无法被删除，必须等Job执行结束后才能成功删除。

更新指定的Pipeline

如下代码可以对指定的pipeline进行更新。

```
 pipelineName := "test"
args, _ := MEDIA_CLIENT.GetPipelineUpdate(pipelineName)
args.Description = "update"
args.TargetBucket = "vwdemo"
args.SourceBucket = "vwdemo"

config := &api.UpdatePipelineConfig{}
config.Capacity = 2
config.Notification = "zz"
args.UpdatePipelineConfig = config
err := MEDIA_CLIENT.UpdatePipeline(pipelineName, args)
if err != nil {
    fmt.Printf("update Pipeline error: %+v\n", err)
    return
}
fmt.Println("update pipeline success")
```

Transcoding-Job转码任务

Transcoding Job(任务)是音视频转码中最基本的执行单元，每个任务将一个原始的音视频资源转码成目标规格的音视频资源。因此，任务和转码的目标是一一对应的，也就是说如果用户需要将一个原始多媒体文件转换成三种目标规格，比如从AVI格式转码成FLV/MP4/HLS格式，那么用户将会需要创建三个任务。

创建Transcoding Job

用户在创建转码任务时，需要为转码任务指定所属的Pipeline、所需应用的Preset以及原始音视频资源的BOS Key以及目标音视频资源BOS Key。

如下代码创建一个Job, 并获取新创建的jobID：

```
 pipelineName := "go_sdk_test"
sourceKey := "test.mp4"
targetKey := "test-result.mp4"
presetName := "test_preset"
jobResponse, err := MEDIA_CLIENT.CreateJob(pipelineName, sourceKey, targetKey, presetName)
if err != nil {
    fmt.Printf("create job error: %+v\n", err)
    return
}
fmt.Println("create job success jobId:", jobResponse.JobId)
```

如下代码创建一个支持视频合并、去水印、加水印（Job上而不是Preset上指定watermarkId）的Job, 并获取新创建的jobID：

```
  args := &api.CreateJobArgs{}
args.PipelineName = "go_sdk_test"
source := &api.Source{Clips: &[]api.SourceClip{{
  SourceKey:              "01.mp4",
  EnableDelogo:           false,
  DurationInMillisecond: 6656,
  StartTimeInSecond:      2}}}
args.Source = source
target := &api.Target{}
targetKey := "clips_playback_watermark_delogo_crop2.mp4"
watermarkId := "wmk-xxxx"
target.TargetKey = targetKey
watermarkIdSlice := append(target.WatermarkIds, watermarkId)
target.WatermarkIds = watermarkIdSlice
presetName := "go_test_customize_audio_video"
target.PresetName = presetName

delogoArea := &api.Area{}
delogoArea.X = 10
delogoArea.Y = 10
delogoArea.Width = 30
delogoArea.Height = 40
target.DelogoArea = delogoArea

args.Target = target

jobResponse, err := MEDIA_CLIENT.CreateJobCustomize(args)
if err != nil {
    fmt.Printf("create job error: %+v\n", err)
    return
}
fmt.Println("create job success jobId:", jobResponse.JobId)
```

列出指定Pipeline的所有Transcoding Job

如下代码通过指定pipelineName查询该Pipeline下的所有Job：

```
  pipelineName := "test"
listTranscodingJobsResponse, err := MEDIA_CLIENT.ListTranscodingJobs(pipelineName)
if err != nil {
    fmt.Printf("list job  error: %+v\n", err)
    return
}
fmt.Printf("list job success : %+v\n", listTranscodingJobsResponse)
```

查询指定的Transcoding Job信息

可以通过如下代码通过jobId读取某个Job：

```
  jobId := "job-xxxxxxxxx"
getTranscodingJobResponse, err := MEDIA_CLIENT.GetTranscodingJob(jobId)
if err != nil {
    fmt.Printf("get job error: %+v\n", err)
    return
}
fmt.Printf("get job success : %+v\n", getTranscodingJobResponse)
```

Preset模板

模板是系统预设的对于一个视频资源在做转码计算时所需定义的集合。用户可以更简便的将一个模板应用于一个和多个视频的转码任务，以使这些任务输出相同规格的目标视频资源。

音视频转码为用户预设了丰富且完备的系统模板，以满足用户对于目标规格在格式、码率、分辨率、加解密、水印等诸多方向上的普遍需求，对于不希望过多了解音视频复杂技术背景的用户来说，是最佳的选择。百度为那些在音视频技术上有着丰富积累的用户，提供了可定制化的转码模板，以帮助他们满足复杂业务条件下的转码需求。

当用户仅需对于音视频的容器格式做变化时，百度提供Transmux（转封装，封装格式转换，比如mp4转mov）模板帮助用户以秒级的延迟快速完成容器格式的转换，比如从MP4转换成HLS，而保持原音视频的属性不变。

查询当前用户Preset及所有系统Preset

用户可以通过如下代码查询所有的Preset

```
 listPresetsResponse, err := MEDIA_CLIENT.ListPresets()
if err != nil {
    fmt.Printf("list preset error: %+v\n", err)
    return
}
fmt.Printf("list preset success: %+v\n", listPresetsResponse)
```

查询指定的Preset信息

如下代码通过指定pipelineName查询该Pipeline下的所有Job：

```
 presetName := "test"
getPresetResponse, err := MEDIA_CLIENT.GetPreset(preset)
if err != nil {
    fmt.Printf("list preset error: %+v\n", err)
    return
}
fmt.Printf("list preset success: %+v\n", getPresetResponse)
```

创建Preset

如果系统预设的Preset无法满足用户的需求，用户可以自定义自己的Preset。根据不同的转码需求，可以使用不同的接口创建Preset。

创建仅支持容器格式转换的Preset

如下代码创建仅执行容器格式转换Preset

```
 presetName := "test"
description := "测试创建模板"
container := "mp4"
err := MEDIA_CLIENT.CreatePreset(presetName, description, container)
if err != nil {
    fmt.Printf("create preset error: %+v\n", err)
    return
}
fmt.Println("create preset success")
```

创建音频文件的转码Preset，不需要截取片段和加密

如果创建一个不需要截取片段和加密的音频文件转码Preset，可以参考如下代码

```
  preset := &api.Preset{}
preset.PresetName = "go_test_customize"
preset.Description = "自定义创建模板"
preset.Container = "mp3"

audio := &api.Audio{}
audio.BitRateInBps = 256000
preset.Audio = audio

err := MEDIA_CLIENT.CreatePrestCustomize(preset)
if err != nil {
    fmt.Printf("create preset error: %+v\n", err)
    return
}
fmt.Println("create preset success")
```

创建音频文件转码Preset，需要设置片段截取属性和加密属性

如果创建一个支持截取片段和加密的音频文件转码Preset，可以参考如下代码

```
  preset := &api.Preset{}
preset.PresetName = "go_test_customize_encryption_clip"
preset.Description = "自定义创建模板"
preset.Container = "mp3"

audio := &api.Audio{}
audio.BitRateInBps = 256000
preset.Audio = audio

clip := &api.Clip{}
clip.StartTimeInSecond = 2
clip.DurationInSecond = 10
preset.Clip = clip

encryption := &api.Encryption{}
encryption.Strategy = "PlayerBinding"
preset.Encryption = encryption

err := MEDIA_CLIENT.CreatePrestCustomize(preset)
err := MEDIA_CLIENT.CreatePrestCustomize(preset)
if err != nil {
    fmt.Printf("create preset error: %+v\n", err)
    return
}
fmt.Println("create preset success")
```

创建视频文件转码Preset，不需要截取片段、加密和水印属性

如果创建一个不需要截取片段，加密和水印的视频文件转码Preset，可以参考如下代码

```
  preset := &api.Preset{}
preset.PresetName = "go_test_customize_audio_video"
preset.Description = "自定义创建模板"
preset.Container = "mp4"

audio := &api.Audio{}
audio.BitRateInBps = 256000
preset.Audio = audio

video := &api.Video{}
video.BitRateInBps = 1024000
preset.Video = video

err := MEDIA_CLIENT.CreatePrestCustomize(preset)
if err != nil {
    fmt.Printf("create preset error: %+v\n", err)
    return
}
fmt.Println("create preset success")
```

创建视频文件转码Preset，需要设置片段截取、加密和水印属性

如果创建一个需要截取片段，加密和添加水印的视频文件转码Preset，可以参考如下代码

```
  preset := &api.Preset{}
preset.PresetName = "go_test_customize_clp_aud_vid_en_wat"
preset.Description = "自定义创建模板"
preset.Container = "mp4"

clip := &api.Clip{}
clip.StartTimeInSecond = 0
clip.DurationInSecond = 60
preset.Clip = clip

audio := &api.Audio{}
audio.BitRateInBps = 256000
preset.Audio = audio

video := &api.Video{}
video.BitRateInBps = 1024000
preset.Video = video

encryption := &api.Encryption{}
encryption.Strategy = "PlayerBinding"
preset.Encryption = encryption

preset.WatermarkID = "wmk-xxxxxx"

err := MEDIA_CLIENT.CreatePrestCustomize(preset)
if err != nil {
    fmt.Printf("create preset error: %+v\n", err)
    return
}
fmt.Println("create preset success")
```

创建Preset，指定所有的参数

如果需要定制所有配置参数，可以参考如下代码

```
 preset := &api.Preset{}
preset.PresetName = "go_test_customize_full_args"
preset.Description = "全参数"
preset.Container = "hls"
preset.Transmux = false

clip := &api.Clip{}
clip.StartTimeInSecond = 0
clip.DurationInSecond = 60
preset.Clip = clip

audio := &api.Audio{}
audio.BitRateInBps = 256000
preset.Audio = audio

video := &api.Video{}
video.BitRateInBps = 1024000
preset.Video = video

encryption := &api.Encryption{}
encryption.Strategy = "PlayerBinding"
preset.Encryption = encryption

water := &api.Watermarks{}
water.Image = []string{"wmk-pc0rdhzbm8ff99qw"}
preset.Watermarks = water

transCfg := &api.TransCfg{}
transCfg.TransMode = "normal"
preset.TransCfg = transCfg

extraCfg := &api.ExtraCfg{}
extraCfg.SegmentDurationInSecond = 6.66
preset.ExtraCfg = extraCfg

err := MEDIA_CLIENT.CreatePrestCustomize(preset)
if err != nil {
    fmt.Printf("create preset error: %+v\n", err)
    return
}
fmt.Println("create preset success")
```

更新Preset

用户可以根据模板名更新自己创建的模板：

```
 preset, _ := MEDIA_CLIENT.GetPreset("go_test_customize")
preset.Description = "test update preset"
err := MEDIA_CLIENT.UpdatePreset(preset)
if err != nil {
    fmt.Printf("update preset error: %+v\n", err)
    return
}
fmt.Println("update preset success")
```

Mediainfo媒体信息

对于BOS中某个Object，可以通过下面代码获取媒体信息

```
 bucket := "bucekt"
key := "key"
info, err := MEDIA_CLIENT.GetMediaInfoOfFile(bucket, key)
if err != nil {
    fmt.Printf("get media information error: %+v\n", err)
    return
}
fmt.Printf("get media information success: %+v\n", info)
```

Thumbnail-Job缩略图任务

缩略图是图片、视频经压缩方式处理后的小图。因其小巧，加载速度非常快，故用于快速浏览。缩略图任务可用于为BOS中的多媒体资源创建缩略图。

创建Thumbnail Job

通过pipeline，BOS Key以及其他配置信息为指定媒体生成缩略图，并获取返回的缩略图任务jobId。可以参考如下代码：

```
 pipelineName := "go_test"
sourcekey := "01.mp4"
target := &api.ThumbnailTarget{}
target.Format = "jpg"
target.SizingPolicy = "keep"
capture := &api.ThumbnailCapture{}
capture.Mode = "manual"
capture.StartTimeInSecond = 0.0
capture.EndTimeInSecond = 5.0
capture.IntervalInSecond = 1.0
createJobResponse, err := MEDIA_CLIENT.CreateThumbnailJob(pipelineName, sourcekey, TargetOp(target),
CaptureOp(capture))
if err != nil {
    fmt.Printf("create thumbanil job error: %+v\n", err)
    return
}
fmt.Println("create thumbanil job success jobId: ", createJobResponse.JobId)
```

创建去水印的缩略图，可以参考如下代码：

```
 pipelineName := "go_test"
sourcekey := "01.mp4"
target := &api.ThumbnailTarget{}
target.KeyPrefix = "taget_key_prefix_test_delogo3"
delogo := &api.Area{}
delogo.X = 20
delogo.Y = 20
delogo.Height = 50
delogo.Width = 80

createJobResponse, err := MEDIA_CLIENT.CreateThumbnailJob(pipelineName, sourcekey, TargetOp(target),
DelogoAreaOp(delogo))
if err != nil {
    fmt.Printf("create thumbanil job error: %+v\n", err)
    return
}
fmt.Println("create thumbanil job success jobId: ", createJobResponse.JobId)
```

创建去水印、去黑边的缩略图，可以参考如下代码：

```
 pipelineName := "go_test"
sourcekey := "01.mp4"
target := &api.ThumbnailTarget{}
target.KeyPrefix = "taget_key_prefix_test_delogo_crop"
delogo := &api.Area{}
delogo.X = 20
delogo.Y = 20
delogo.Height = 50
delogo.Width = 80

crop := &api.Area{}
crop.X = 120
crop.Y = 120
crop.Height = 100
crop.Width = 80

createJobResponse, err := MEDIA_CLIENT.CreateThumbnailJob(pipelineName, sourcekey,
                                        TargetOp(target), DelogoAreaOp(delogo), CropOp(crop))
if err != nil {
    fmt.Printf("create thumbanil job error: %+v\n", err)
    return
}
fmt.Println("create thumbanil job success jobId: ", createJobResponse.JobId)
```

创建去水印缩略图任务，其中指定了缩略图格式为jpg、尺寸为与原视频保持一致（keep），抽帧模式（SizingPolicy）为split，根据指定的起止时间和张数截取缩略图，FrameNumber则指定了缩略图张数，代码如下：

```
 pipelineName := "go_test"
sourcekey := "01.mp4"
target := &api.ThumbnailTarget{}
target.Format = "jpg"
target.SizingPolicy = "keep"

capture := &api.ThumbnailCapture{}
capture.Mode = "split"
capture.FrameNumber = 30

delogo := &api.Area{}
delogo.X = 20
delogo.Y = 20
delogo.Height = 50
delogo.Width = 80

createJobResponse, err := MEDIA_CLIENT.CreateThumbnailJob(pipelineName, sourcekey,
                                        TargetOp(target), CaptureOp(capture), DelogoAreaOp(delogo))
if err != nil {
    fmt.Printf("create thumbanil job error: %+v\n", err)
    return
}
fmt.Println("create thumbanil job success jobId: ", createJobResponse.JobId)
```

如果只想创建一个简单的缩略图任务可以参考如下代码：

```
 pipelineName := "go_test"
sourcekey := "01.mp4"
createJobResponse, err := MEDIA_CLIENT.CreateThumbnailJob(pipelineName, sourcekey)
if err != nil {
    fmt.Printf("create thumbanil job error: %+v\n", err)
    return
}
fmt.Println("create thumbanil job success jobId: ", createJobResponse.JobId)
```

查询指定Thumbnail Job

如果需要获取一个已创建的缩略图任务的信息，可以参考如下代码：

```
 jobId := "job-xxxxxxx"
jobResponse, err := MEDIA_CLIENT.GetThumbanilJob(jobId)
if err != nil {
    fmt.Printf("get thumbanil job error: %+v\n", err)
    return
}
fmt.Printf("get thumbanil job success job: %+v\n", jobResponse)
```

查询指定队列的Thumbnail Jobs

如果需要获取一个队列里的全部缩略图任务的信息，可以参考如下代码：

```
 pipelineName := "go_sdk_test"
listThumbnailJobsResponse, err := MEDIA_CLIENT.ListThumbnailJobs(pipelineName)
if err != nil {
    fmt.Printf("list thumbanil job error: %+v\n", err)
    return
}
for _, job := range listThumbnailJobsResponse.Thumbnails {
  fmt.Printf("list thumbanil job success : %+v\n", job)
}
```

Watermark水印

数字水印是向数据多媒体（如图像、音频、视频信号等）中添加某些数字信息以达到文件真伪鉴别、版权保护等功能。嵌入的水印信息隐藏于宿主文件中，不影响原始文件的可观性和完整性。

用户可以将BOS中的一个Object创建为水印，获得对应的watermarkId。然后在转码任务中将此水印添加到目的多媒体文件。

创建水印

如果需要创建一个水印，指定水印的位置，并获得水印的唯一ID（其中bucket是水印文件所在bucket名称，key是水印文件在该bucket中的文件名），可以参考如下代码：

```
 args := &api.CreateWaterMarkArgs{}
args.Bucket = "go-test"
args.Key = "01.jpg"
args.HorizontalAlignment = "right"
args.VerticalAlignment = "top"
createWaterMarkResponse, err := MEDIA_CLIENT.CreateWaterMark(args)
if err != nil {
    fmt.Printf("create watermark job error: %+v\n", err)
    return
}
fmt.Println("create watermark job success Id: ", createWaterMarkResponse.WatermarkId)
```

如果需要创建一个水印，指定水印的位置、显示时间段、重复显示次数（动态水印）、自动缩放，并获得水印的唯一ID，可以参考如下代码：

```
 args := &api.CreateWaterMarkArgs{}
args.Bucket = "go-test"
args.Key = "01.jpg"
args.HorizontalAlignment = "left"
args.VerticalAlignment = "top"
args.HorizontalOffsetInPixel = 20
args.VerticalOffsetInPixel = 10
timeline := &api.Timeline{}
timeline.StartTimeInMillisecond = 1000
timeline.DurationInMillisecond = 3000
args.Timeline = timeline
args.Repeated = 1
args.AllowScaling = true
createWaterMarkResponse, err := MEDIA_CLIENT.CreateWaterMark(args)
if err != nil {
    fmt.Printf("create watermark job error: %+v\n", err)
    return
}
fmt.Println("create watermark job success Id: ", createWaterMarkResponse.WatermarkId)
```

查询指定水印

如果需要查询已创建的水印，可以参考如下代码：

```
 waterMarkId := "wmk-xxx"
response, err := MEDIA_CLIENT.GetWaterMark(waterMarkId)
if err != nil {
    fmt.Printf("get watermark job error: %+v\n", err)
    return
}
fmt.Printf("get watermark job success: %+v\n", response)
```

查询当前用户水印

如果需要查询出本用户所创建的全部水印，可以参考如下代码：

```
 response, err := MEDIA_CLIENT.ListWaterMark()
if err != nil {
    fmt.Printf("get watermark job error: %+v\n", err)
    return
}
for _, watermark := range response.Watermarks {
  fmt.Printf("watermark job: %+v\n", watermark)
}
```

删除水印

如果需要删除某个已知watermarkId的水印，可以参考如下代码：

```
 waterMarkId := "wmk-xxx"
err := MEDIA_CLIENT.DeleteWaterMark(waterMarkId)
if err != nil {
    fmt.Printf("delete watermark job error: %+v\n", err)
    return
}
fmt.Println("delete watermark success")
```

Notification通知

通知功能可以在音视频转码任务状态转换时主动向开发者服务器推送消息。

创建通知

如果需要创建通知可以参考如下代码：

```
 name := "test"
endpoint := "http://www.baidu.com"
err := MEDIA_CLIENT.CreateNotification(name, endpoint)
if err != nil {
    fmt.Printf("create notification error: %+v\n", err)
    return
}
fmt.Println("create notification success")
```

查询指定通知

如果需要查询已创建的通知，可以参考如下代码：

```
 name := "test"
response, err := MEDIA_CLIENT.GetNotification(test)
if err != nil {
    fmt.Printf("get notification error: %+v\n", err)
    return
}
fmt.Printf("get notification success : %+v\n", response)
```

查询当前用户通知

如果需要查询出本用户所创建的全部通知，可以参考如下代码：

```
 response, err := MEDIA_CLIENT.ListNotification()
if err != nil {
    fmt.Printf("list user`s notification error: %+v\n", err)
    return
}
for _, notification := range response.Notifications {
 fmt.Printf("list notification success : %+v\n", notification)
}
```

删除通知

如果需要删除某个通知，可以参考如下代码：

```
  name := "test"
err := MEDIA_CLIENT.DeleteNotification(name)
if err != nil {
    fmt.Printf("delete notification error: %+v\n", err)
    return
}
fmt.Println("delete notification success")
```

错误处理

GO语言以error类型标识错误，MCP支持两种错误见下表：

错误类型	说明
BceClientError	用户操作产生的错误
BceServiceError	MCP服务返回的错误

用户使用SDK调用MCP相关接口，除了返回所需的结果之外还会返回错误，用户可以获取相关错误进行处理。实例如下：

```
  // MEDIA_CLIENT 为已创建的MCP Client对象
result, err := MEDIA_CLIENT.ListPipelines()
if err != nil {
   switch realErr := err.(type) {
   case *bce.BceClientError:
      fmt.Println("client occurs error:", realErr.Error())
   case *bce.BceServiceError:
      fmt.Println("service occurs error:", realErr.Error())
   default:
      fmt.Println("unknown error:", err)
   }
}
```

客户端异常

客户端异常表示客户端尝试向MCP发送请求以及数据传输时遇到的异常。例如，当发送请求时网络连接不可用时，则会返回BceClientError。

服务端异常

当MCP服务端出现异常时，MCP服务端会返回给用户相应的错误信息，以便定位问题。常见服务端异常可参见MCP错误码

SDK日志

MCP GO SDK支持六个级别、三种输出（标准输出、标准错误、文件）、基本格式设置的日志模块，导入路径为`github.com/baidubce/bce-sdk-go/util/log`。输出为文件时支持设置五种日志滚动方式（不滚动、按天、按小时、按分钟、按大小），此时还需设置输出日志文件的目录。

默认日志

MCP GO SDK自身使用包级别的全局日志对象，该对象默认情况下不记录日志，如果需要输出SDK相关日志需要用户自定指定输出方式和级别，详见如下示例：

```
// import "github.com/baidubce/bce-sdk-go/util/log"

// 指定输出到标准错误，输出INFO及以上级别
log.SetLogHandler(log.STDERR)
log.SetLogLevel(log.INFO)

// 指定输出到标准错误和文件，DEBUG及以上级别，以1GB文件大小进行滚动
log.SetLogHandler(log.STDERR | log.FILE)
log.SetLogDir("/tmp/gosdk-log")
log.SetRotateType(log.ROTATE_SIZE)
log.SetRotateSize(1 << 30)

// 输出到标准输出，仅输出级别和日志消息
log.SetLogHandler(log.STDOUT)
log.SetLogFormat([]string{log.FMT_LEVEL, log.FMT_MSG})
```

说明：

1. 日志默认输出级别为`DEBUG`
2. 如果设置为输出到文件，默认日志输出目录为`/tmp`，默认按小时滚动
3. 如果设置为输出到文件且按大小滚动，默认滚动大小为1GB
4. 默认的日志输出格式为：`FMT_LEVEL, FMT_LTIME, FMT_LOCATION, FMT_MSG`

项目使用

该日志模块无任何外部依赖，用户使用GO SDK开发项目，可以直接引用该日志模块自行在项目中使用，用户可以继续使用GO SDK使用的包级别的日志对象，也可创建新的日志对象，详见如下示例：

```
// 直接使用包级别全局日志对象（会和GO SDK自身日志一并输出）
log.SetLogHandler(log.STDERR)
log.Debugf("%s", "logging message using the log package in the MCP go sdk")

// 创建新的日志对象（依据自定义设置输出日志，与GO SDK日志输出分离）
myLogger := log.NewLogger()
myLogger.SetLogHandler(log.FILE)
myLogger.SetLogDir("/home/log")
myLogger.SetRotateType(log.ROTATE_SIZE)
myLogger.Info("this is my own logger from the MCP go sdk")
```

版本变更记录

首次发布:
202-04-23

- MCP支持go-sdk啦，现在您可以通过golang调用MCP-SDK服务。当前SDK能力支持pipeline队列操作、Transcoding-Job转码任务操作、Preset模板操作、Thumbnail-Job缩略图任务操作、Watermark水印任务操作、MediaInfo媒资信息操作、Notification通知操作。

典型实践

视频版权保护

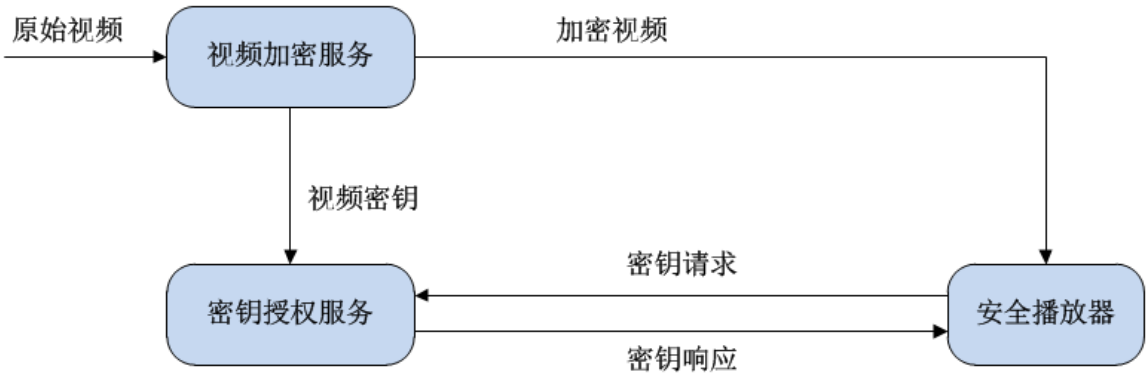
概述

百度智能云视频版权保护服务是一种安全易用的轻量级版权保护服务，通过视频转码平台用AES128加密算法对视频文件进行加密，防止非法用户对视频内容进行复制和扩散。

目前版权保护服务仅针对HLS格式的视频提供保护，对于其他格式的视频，须先转码为HLS，再进行加密处理。

版权保护架构

视频版权保护架构包含视频加密服务、密钥授权服务和安全播放器三部分，如下图所示：



- **视频加密服务**

  在视频转码过程中使用AES128对转码HLS格式的视频进行加密，视频加密封装符合HLS规范。对于非HLS格式的视频，视频加密服务须与视频转码服务配合使用，先将输入视频转码为HLS格式，再进行加密。

- **密钥授权服务**

  提供视频解密密钥授权服务，合法用户可使用安全播放器获取解密密钥并解密播放视频内容。

  密钥授权包括Open和PlayerBinding两种模式：

  - Open模式：解密密钥面向所有用户。
  - PlayerBinding模式：密钥访问有权限控制，非授权用户不能获取解密密钥。
  - Token模式： 密钥访问增加Token验证，提供更安全的访问控制。

- **安全播放器**

  加密后的视频不能用通用播放器播放，须使用百度智能云提供的安全播放器，您可以基于百度智能云播放器SDK定制Web/Android/iOS安全播放器。

版权保护操作流程

视频加密

1. 登录百度智能云官网。
   - 未注册，须先注册百度智能云账户。
   - 已注册，直接登录。
2. 选择“产品服务>音视频转码MCP”，在左侧侧边栏中选择“视频转码”--“转码模板”，在右侧“转码模板”列表页面进行如下操作：

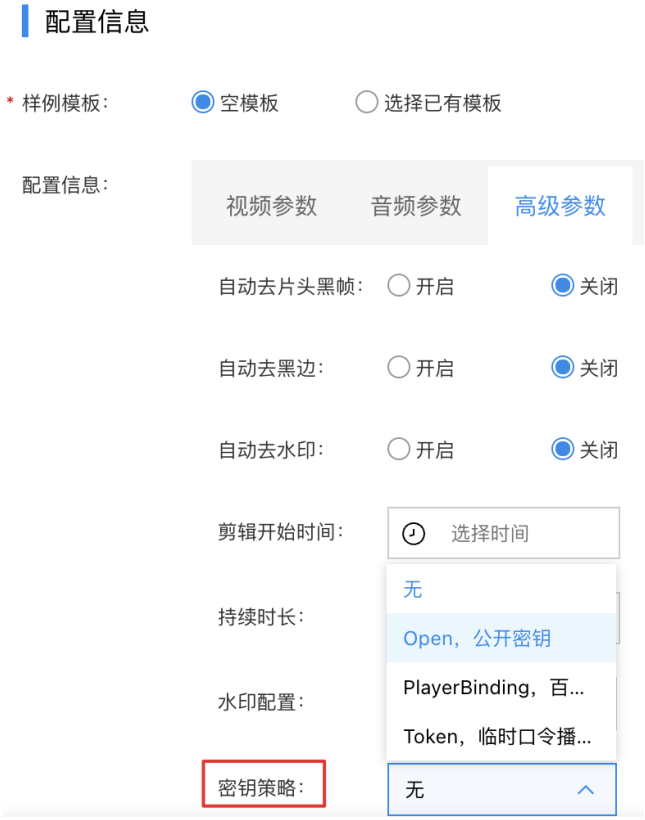
   (1) 点击“新建普通转码模板”。

   (2) 填写基本信息。

   建议您在“音/视频格式”下拉列表中选择HLS格或者A-HLS音视频容器。

   (3) 勾选配置信息。

   勾选以下可配置项：
   - 密钥策略



视频加密配置有以下可选项： - Open：开放密钥，系统自动生成加密密钥，密钥公开，不设访问控制。 - PlayerBinding：绑定播放器，系统自动生成加密密钥，密钥设有访问控制。 PlayerBinding模式下密钥设有访问控制，安全性比较高，推荐使用PlayerBinding模式。 - Token: 临时口令播放授权，系统根据UserKey生成密钥加密视频；播放时按照规则生成Token并发送给密钥服务验证，校验通过才能播放，安全性比较高。

(4) 单击“立即创建”创建完成。

3. 选择“产品服务>对象存储BOS”，新建两个Bucket为创建队列做准备。

   视频转码任务的输入输出均须采用BOS Bucket，您至少需要拥有一个BOS Bucket才能使用相关功能。建议您准备两个BOS Bucket分别用于视频输入与视频输出。

4. 创建队列。

   输入/输出Bucket分别选择步骤3创建的BOS Bucket。

5. 创建转码任务。

   - “队列”选择步骤4创建的队列。
   - “源文件输入”选择存储于输入Bucket中的待加密视频。
   - “转码模板”选择步骤2创建的用户模板。

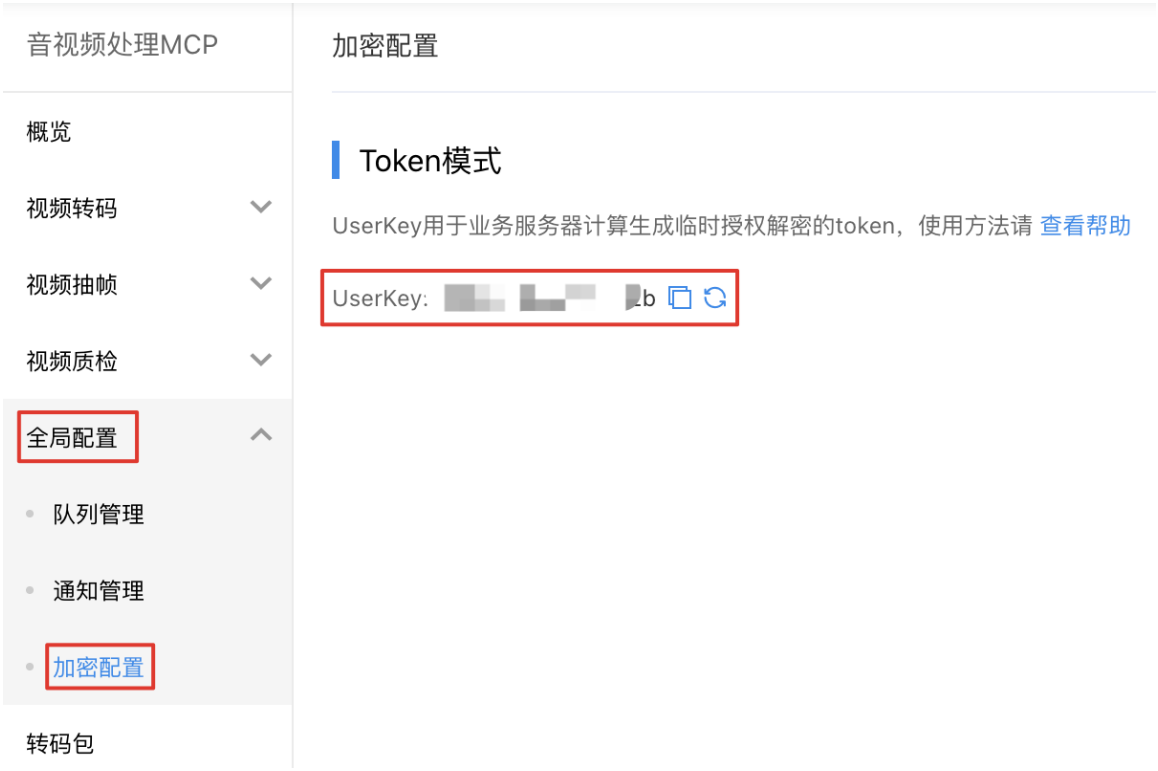
   转码任务的状态显示为“成功”后，即可进入输出Bucket查看加密后的视频。

解密播放视频

- 加密后的音视频不能用普通播放器播放，须使用百度智能云提供的安全播放器播放，播放方法与非加密视频完全相同。
- VOD Token模式详细说明
- Videoworks Token模式详细说明
- MCP解密HLS视频的m3u8文件示例：

```
#EXTM3U
#EXT-X-VERSION:3
#EXT-X-TARGETDURATION:10
#EXT-X-KEY:METHOD=AES-128,URI="https://d...  .../v1/vid
eoKey?videoKeyId=job-kghcgv4tu87fdsd6",IV=0xcc...
#EXTINF:10.000000,
...m3u8.0.ts
```

- MCP获取UserKey说明：



视频添加字幕

概览

通过视频转码的方式给原始视频添加字幕。

需求场景

视频添加字幕

用户需要将无字幕的视频添加字幕。

方案概述

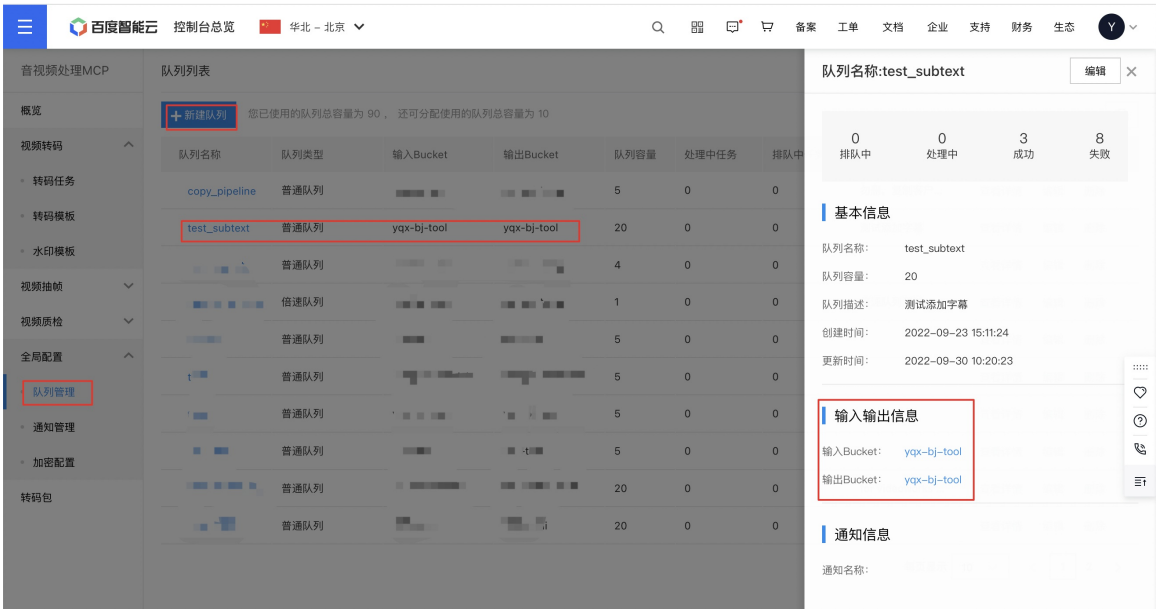
通过MCP的转码功能，使用SRT字幕文件给视频添加字幕流。添加后视频播放时可带上相关字幕内容。

配置准备

- 转码队列：一个用于执行指定转码任务的任务队列
- 转码模板：对原始视频分辨率进行修改、去除黑边、视频格式变化、添加字幕等等功能。详情可查看转码模板接口。

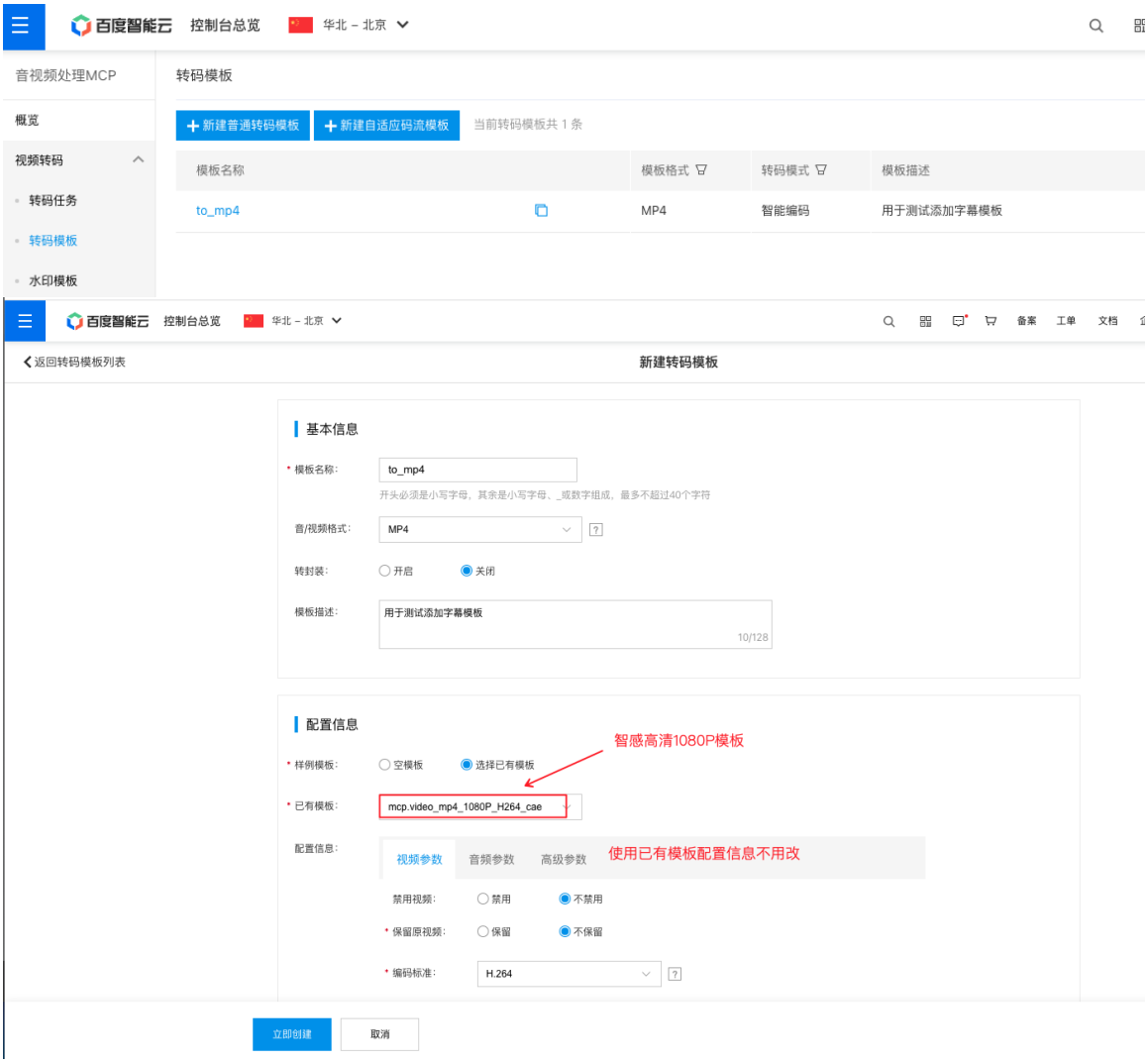
> 说明：本实践中只需要用到转码添加字幕功能，但是也要创建一个转码模板用于嵌入。如果添加字幕同时还有转码需求、则可在一个转码任务中完成。

转码队列 MCP控制台可进行新建，用于控制传入的转码视频文件所在bucket、转码后视频文件保存bucket。如果已有队列满足需求可不用新建，直接使用即可



转码模板 如果已有的模板满足视频转码需求，可使用原有模板。

如不满足可创建一个智感高清1080P的Mp4格式模板。



测试验证

本实践以一个Mp4视频添加测试字幕文件进行示例。

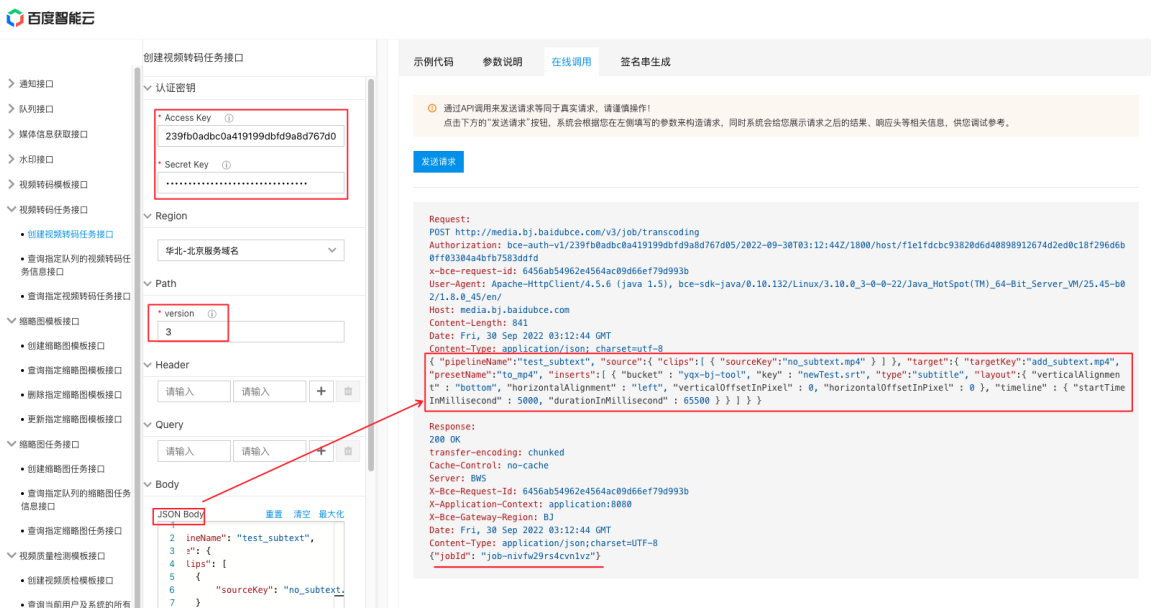
控制台的新增转码任务配置中，并没有添加字幕的功能，只能用API调用的方式进行使用。本实践使用APIExplorer的创建视频转码任务接口进行接口调用。

- 该接口中ak-sk需要自行填写
- version参数固定【3】
- Json body参数参考

```
{
  "pipelineName": "test_subtext",    # 前面创建的转码队列名称
  "source": {
     "clips": [
        {
           "sourceKey": "no_subtext.mp4"  # 保存的待添加字幕文件
        }
     ]
  },
  "target": {
     "targetKey": "add_subtext.mp4",  # 生成的带字幕文件
     "presetName": "to_mp4",          # 创建的转码模板名称
     "inserts": [                     # 添加字幕操作
        {
           "bucket" : "yqx-bj-tool",  # 字幕文件保存bucket
           "key" : "newTest.srt",     # 字幕文件名称
           "type": "subtitle",        # 添加字幕类型
           "layout": {                # 布局
              "verticalAlignment" : "bottom",
              "horizontalAlignment" : "left",
              "verticalOffsetInPixel" : 0,
              "horizontalOffsetInPixel" : 0
           },
           "timeline" : {
              "startTimeInMillisecond" : 1000,
              "durationInMillisecond" : 65500
           }
        }
     ]
  }
}
```

> 说明：更多参数字段含义参考创建视频转码任务接口。

调用示例如下：



注：转码任务执行根据视频大小不等执行耗时不同，调用完成后可使用返回的jobId查看转码进行，控制台查看方式如下

添加完字幕视频效果截图展示

视频专区

功能效果演示

MCP老片修复功能效果演示

- 本视频展示了智感超清老片修复的效果，能去除噪点、马赛克、抖动等，强化边缘纹理细节，修复画面质量。

MCP感知编码功能效果演示

- 本视频展示了智感超清内容自适应转码效果，根据内容复杂度动态分配码率参数，提高视频清晰度的同时，最多可节省50%+带宽成本。

MCP视频超分功能效果演示

- 本视频展示了智感超清超分辨率的效果，将360p提升至1080p，并通过智能插帧技术进行帧率上采样，重建超高清画质。

常见问题

常见问题总览

视频上传

- 音视频转码API支持上传吗？

视频转码

- 在MCP中，如果提交了多个视频转换任务，那么这些视频是一个一个转换，还是并行的转呢？
- MCP支持输入音频+无声视频合成有声视频吗？
- 目前支持输出哪些视频转码格式？
- 如何实现批量转码？
- MCP的转码任务中的jobid可能重复吗?

性能类问题

- 如何提升用户加载视频的速度？

播放器相关

- 手机端web播放器的使用注意事项有哪些？
- 可以使用通用的播放器来播放转码后的视频吗？
- 视频服务器上部署了crossdomain.xml，但flash播放器仍然无法访问服务器上的视频？
- 视频文件放在BOS上，怎样设置crossdomain.xml？
- Web player可以播放mp4视频，但是不能播放m3u8视频？
- Web player在PC上可以正常播放，但是在手持设备上不能播放？

服务等级协议相关

- 免费队列的转码服务，有SLA保证吗？

视频上传

音视频转码API支持上传吗？

音视频转码是一个转码服务，视频存储在对象存储（BOS）上，需要使用云存储（BOS）的API进行上传。

视频转码

在MCP中，如果提交了多个视频转换任务，那么这些视频是一个一个转换，还是并行的转呢？

是并发处理的。MCP中，一个任务队列中可以包含最多20个进行中的转码任务，每个账户可以有5个任务队列。队列总数小于限制时，可以创建新的队列。

音视频转码MCP支持输入音频+无声视频合成有声视频吗？

视频文件支持叠加音频文件，参考视频转码任务接口Job-Transcoding-API中的inserts字段，详见 Job-Transcoding-API。

目前支持输出哪些视频转码格式？

- 支持输出的视频格式为FLV、MP4、HLS。
- 支持输出的音频格式为MP3、M4A。

如何实现批量转码？

您可通过使用服务端的SDK，进行批量转码。具体参考SDK文档

音视频转码MCP的转码任务中的jobid可能重复吗?

MCP转码生成的job id是不会重复的

转码速度很慢，能提速吗？

针对各类用户对转码速度的要求不同，我们提供了三个档次的转码服务：

- 标准转码服务：针对短/小视频，不分片，速度能达到1-3倍速；
- 极速转码服务：针对长视频，进行分片转码，100分钟720P的转码速度能达到10倍速；
- 智能分片转码：针对长视频，根据原视频参数/模板参数/资源情况综合因素，通过AI模型产出智能分片策略，100分钟720p长视频可达50倍速。

转码后画质较差，能提高清晰度吗？

针对不同的内容类型、视频时长以及转码诉求（例如：质量提升优先还是码率节省优先等），转码参数都是不一样，因此根据具体的需求，我们都可以安排技术人员进行专项参数调优，大概周期为1-2个工作日。

音视频转码MCP的转码输出格式是否支持A-HLS?

音视频转码MCP的转码输出格式是支持A-HLS的，在创建转码任务时，选择自定义模板时可以设置为a-hls输出格式。

MCP是否支持直接读取百度网盘数据进行转码？

由于音视频转码MCP与百度网盘并非是同一平台产品，无法直接读取网盘数据，您可以把下载网盘数据上传到对象存储bos中，再通过MCP进行转码。

性能类问题

如何提升用户加载视频的速度？

建议配合使用百度智能云的CDN服务。百度智能云的CDN针对多媒体内容做过深度的优化，具体的操作配置防范参见CDN的操作指南。

视频剪辑编辑功能较弱？

目前支持多段视频的拼接、剪辑、叠加音频、水印（jpg、png、mp4）、字幕（字体、字号、位置）、去logo、去黑边等视频编辑处理。更强的渲染合成能力预计Q3能完成，届时会支持滤镜、转场等各类特效，同时可视化的H5视频编辑工具，9月底会在VOD上线。

智感超清1.0和智感超清2.0的区别？

智感超清1.0的核心技术是内容自适应编码，即根据不同画面复杂度智能分配最优编码参数，主要收益是保证同等画质下尽可

能的节省码率节省带宽成本，基于CPU运行。

智感超清2.0的核心是画质增强，包括：细节增强、色彩增强、明暗度优化、伪影/马赛克等修复等，收益是码率不变大幅提升主观画质，基于GPU运行。

智感超清的模板可以自定义吗？

由于智感超清的很多参数具备专业性，为了保证模板效果，智感超清的模板不可以自定义，我们提供系统模板，若系统模板不能满足需求，您可以联系我们，自定义一个普通模板，将分辨率/码率/帧率等基础参数设置完成后，我们再在后台为您调整为智感超清模板。

将视频中的音频提取出来,以音频文件进行存放,可以实现吗？

目前音视频转码MCP，可以在转码模板中设置“禁止视频”，再通过设置的转码模板进行转码视频，生成对应音频文件。

MCP是否可以把多个ts文件合并为一个ts文件?

目前音视频转码MCP并不支持合并多个ts文件，生成为一个ts文件。

播放器相关

手机端web播放器的使用注意事项有哪些？

手机由于其自生浏览器的限制，手机浏览器web播放与PC浏览器在表现上存在一些区别如下：

1. 手机浏览器不支持自动播放，controls 必须设置为true。
2. 手机浏览器上不支持加密视频的解密播放。
3. 目前主要支持mp4和m3u8这两种主流视频格式，不支持flv和rtmp格式，且部分android手机不支持m3u8文件的播放。
4. 因为兼容性问题，一些功能比如图片广告、右键、跑马灯、视频打点等均不建议在手机上使用。

可以使用通用的播放器来播放转码后的视频吗？

可以。

自定义格式的加密需要用专门的播放器来播放，如有需要，请发工单联系百度技术团队，或加hi群：1464515。

视频服务器上部署了crossdomain.xml，但flash播放器仍然无法访问服务器上的视频？

请检查crossdomain.xml中的授权是否正确，并确认文件的字符编码为UTF-8编码。

视频文件放在BOS上，怎样设置crossdomain.xml？

将crossdomain.xml放在BOS的bucket根目录，并且视频URL采用bucket name前置的方式`https://bucketName.bj.bcebos.com/path/filename`。

另外需要确保如下文件可访问：`https://bucketName.bj.bcebos.com/crossdomain.xml`

Web player可以播放mp4视频，但是不能播放m3u8视频？

请检查crossdomain.xml是否正确设置，m3u8视频是采用flash模式，需要跨域许可，mp4视频优先采用html5模式播放，不需要跨域许可。

Web player在PC上可以正常播放，但是在手持设备上不能播放？

手持设备上只支持mp4格式视频播放。

版权保护的解决方案是什么样的？ 目前仅支持HLS格式的视频加密和百度专有原生播放器的解密，详细方案见视频版权保护。

服务等级协议相关

免费队列的转码服务，有SLA保证吗？

MCP从2019年4月1日起，所有转码服务均由免费改为了收费，不再有免费队列，所有的转码服务均有SLA，详见 MCP服务等级协议SLA。

服务等级协议SLA

MCP服务等级协议SLA

协议生效时间：2019年4月1日

本服务等级协议（Service Level Agreement，简称“SLA”） 规定了百度智能云向客户提供的音视频处理服务MCP（Multimedia Cloud Processing，以下简称"MCP"）的服务可用性等级指标及赔偿方案。

1.定义

服务周期： 一个服务周期为一个自然月。

服务范围： 中国大陆地区（不含港澳台地区）。

服务周期总分钟数： 按照每月每周七（7）天每天二十四（24）小时计算。

错误请求： MCP将HTTP状态码为5XX和因为MCP服务故障导致的用户正常请求未能到达MCP服务器端的请求视为错误请求。

有效请求： 客户在百度智能云服务账号下所有的MCP媒体转码处理服务，在MCP服务器收到的请求视为有效请求。

每5分钟错误率：

$$每5分钟错误率 = \frac{每5分钟错误请求数}{每5分钟总有效请求数} \times 100\%$$

月度服务费用： 客户在一个自然月中就MCP服务所支付的服务费用。

2. 服务可用性

2.1 服务可用性计算方式

MCP服务可用性按服务周期统计，根据服务周期内每5分钟错误率之和除以服务周期内5分钟的总个数计算出每5分钟错误率的平均值，从而计算得出服务可用性，即：

$$服务可用性 = \left(1 - \frac{服务周期内\Sigma\ 每5分钟错误率}{服务周期内5分钟总个数}\right) \times 100\%$$

注：服务周期内5分钟总个数=12 *24 *该服务周期的天数

2.2 服务可用性承诺

MCP服务可用性承诺在一个周期内不低于99.90%，如MCP服务未达到上述服务可用性承诺，客户可以根据本协议第3条约定获得赔偿。赔偿范围不包括以下原因所导致的请求失败或服务不可用：

(1) 百度智能云预先通知用户后进行系统维护所引起的，包括合理升级、变更、停机、割接、维修和模拟故障演练；

(2) 任何百度智能云所属设备以外的网络、设备故障或配置调整引起的；

(3) 用户内容违规或其他原因导致域名被封禁而产生的错误；

(4) 用户大规模流量突发增长未提前书面告知百度智能云所导致的可用性降低；

(5) 用户的应用程序或数据信息受到黑客攻击而引起的；

(6) 用户维护不当或保密不当致使数据、口令、密码等丢失或泄漏所引起的；

(7) 用户的疏忽或由用户授权的操作所引起的；

(8) 非中国大陆内的请求所导致的错误；

(9) 不可抗力以及意外事件引起的；

(10) 其他非百度智能云原因所造成的不可用。

3. 赔偿方案

3.1 赔偿标准

根据客户在百度智能云账号下MCP的月度服务可用性，按照下表中的标准计算赔偿金额。赔偿方式仅限于用于支付MCP产品的代金券，且赔偿总额不超过未达到服务可用性承诺的当月客户就该账号下MCP支付的月度服务费用的50%。

服务可用性	**赔付代金券金额**
低于99.90%但是等于或高于99%	月度服务费的10%
低于99%但等于或高于95%	月度服务费的25%
低于95%	月度服务费的50%

3.2 赔偿申请时限

客户可以在每月第五（5）个工作日后对上个月没有达到可用性的服务提出赔偿申请。赔偿申请必须限于在MCP没有达到可用性的相关月份结束后两（2）个月内提出。超出申请时限的赔偿申请将不被受理。

4. 其他说明

(1) 在法律法规允许的范围内，百度智能云负责对本协议进行解释说明。

(2) 本协议一经公布立即生效，百度智能云有权对本SLA条款作出修改。如本SLA条款有任何修改，百度智能云以网站公示或发送邮件的方式通知您。如您不同意百度 云对SLA所做的修改，您有权停止使用MCP服务，如您继续使用MCP服务，则视为您接受修改后的SLA。

(3) 本协议项下百度智能云MCP服务对于用户所有的通知均可通过网页公告、站内信、电子邮件、手机短信或其他形式等方式进行；该类通知于发送之日视为已送达收件人。百度智能云不对用户承担因此产生的任何损失。

(4) 本协议的订立、执行和解释及争议的解决均应适用中国法律并受中国法院管辖。如双方就本协议内容或其执行发生任何争议，双方应尽量友好协商解决；协商不成时，任何一方均可向北京市海淀区人民法院提起诉讼。

(5) 本协议构成双方对本协议之约定事项及其他有关事宜的完整协议，除本协议规定的之外，未赋予本协议各方其他权利。

(6) 如本协议中的任何协议无论因何种原因完全或部分无效或不具有执行力，本协议的其余协议仍应有效并且有约束力。

(7) 关于用户约束条款，详见《用户服务协议》中的"用户的权利与义务"相关条款内容。

(8) 关于服务商免责条款，详见《用户服务协议》的"免责声明"相关条款内容。