数据工坊(DWR)

用户指南

文档版本	01
发布日期	2024-07-16

华为技术有限公司

商标声明


和其他华为商标均为华为技术有限公司的商标。
本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

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

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

安全声明

漏洞处理流程

华为公司对产品漏洞管理的规定以“漏洞处理流程”为准，该流程的详细内容请参见如下网址：
https://www.huawei.com/cn/psirt/vul-response-process
如企业客户须获取漏洞信息，请参见如下网址：
https://securitybulletin.huawei.com/enterprise/cn/security-advisory

目 录

1 权限管理

1.1 创建用户并授权使用 DWR

如果您需要对您所拥有的DWR服务进行精细的权限管理，您可以使用**统一身份认证服务**（Identity and Access Management，简称IAM），通过IAM，您可以：

- 根据企业的业务组织，在您的华为云账号中，给企业中不同职能部门的员工创建IAM用户，让员工拥有唯一安全凭证，并使用DWR资源。
- 根据企业用户的职能，设置不同的访问权限，以达到用户之间的权限隔离。
- 将DWR资源委托给更专业、高效的其他华为云账号或者云服务，这些账号或者云服务可以根据权限进行代运维。

如果华为云账号已经能满足您的要求，不需要创建独立的IAM用户，您可以跳过本章节，不影响您使用DWR服务的其它功能。

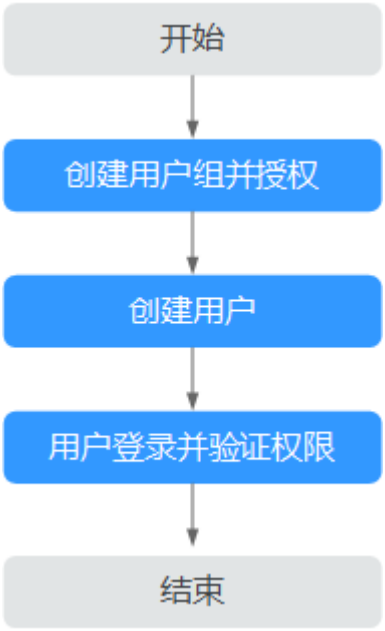
本章节为您介绍对用户授权的方法，操作流程如**图1-1**所示。

前提条件

给用户组授权之前，请您了解用户组可以添加的DWR权限，并结合实际需求进行选择，DWR支持的系统权限，请参见**DWR系统权限**。

示例流程

图 1-1 给用户授予 DWR 权限流程



1. **创建用户组并授权**
   在IAM控制台创建用户组，并授予数据工坊只读权限“DWR ReadOnlyAccess”。
2. **创建用户并加入用户组**
   在IAM控制台创建用户，并将其加入**1**中创建的用户组。
3. **用户登录**并验证权限
   新创建的用户登录控制台，切换至授权区域，验证权限：在“服务列表”中选择数据工坊，进入DWR工作流页面，单击右上角“创建工作流”，尝试创建工作流，如果无法创建（假设当前权限仅包含DWR ReadOnlyAccess），表示“DWR ReadOnlyAccess”已生效。

1.2 DWR 自定义策略

如果系统预置的DWR权限，不满足您的授权要求，可以创建自定义策略。

目前华为云支持以下两种方式创建自定义策略：

- 可视化视图创建自定义策略：无需了解策略语法，按可视化视图导航栏选择云服务、操作、资源、条件等策略内容，可自动生成策略。
- JSON视图创建自定义策略：可以在选择策略模板后，根据具体需求编辑策略内容；也可以直接在编辑框内编写JSON格式的策略内容。
  具体创建步骤请参见**创建自定义策略**。本章为您介绍常用的DWR自定义策略样例。

DWR 自定义策略样例

- 示例1：授权用户创建、删除、执行工作流

```
{
 "Version": "1.1",
 "Statement": [{
  "Effect": "Allow",
  "Action": [
        "dwr:workflow:get*",
        "dwr:workflow:list*",
        "dwr:workflow:createWorkflow",
        "dwr:workflow:deleteWorkflow",
        "dwr:workflow:executeAsync",
  ],
 }]
}
```

- 示例2：授权用户查询、创建、删除、禁用第三方算子

```
{
 "Version": "1.1",
 "Statement": [{
  "Effect": "Allow",
  "Action": [
       "dwr:workflow:get*",
       "dwr:workflow:list*",
       "dwr:workflow:createMyActionTemplate",
       "dwr:workflow:deleteMyActionTemplate",
       "dwr:workflow:forbidMyActionTemplate",
  ],
 }]
}
```

2 算子管理

2.1 算子市场介绍

算子市场即为DWR提供的算子库，提供方包括华为和第三方。

- 华为云自有算子的能力源是华为云数据处理相关的云服务，如媒体处理MPC、图像识别Image等，DWR将云服务提供的各种数据处理能力通过函数生成算子并在算子市场发布。
- 第三方算子是基于DWR的算子注册能力，由第三方开发者创建，专业人员审核发布的公共算子，您也可以将自己创建的算子发布为第三方算子，开放给所有华为云用户使用，详见**发布算子**。

图 2-1 算子市场

2.2 发布算子

操作场景

所有开发者均可以创建用于数据处理的算子，审核通过后发布至DWR算子市场，开放给所有华为云用户使用。

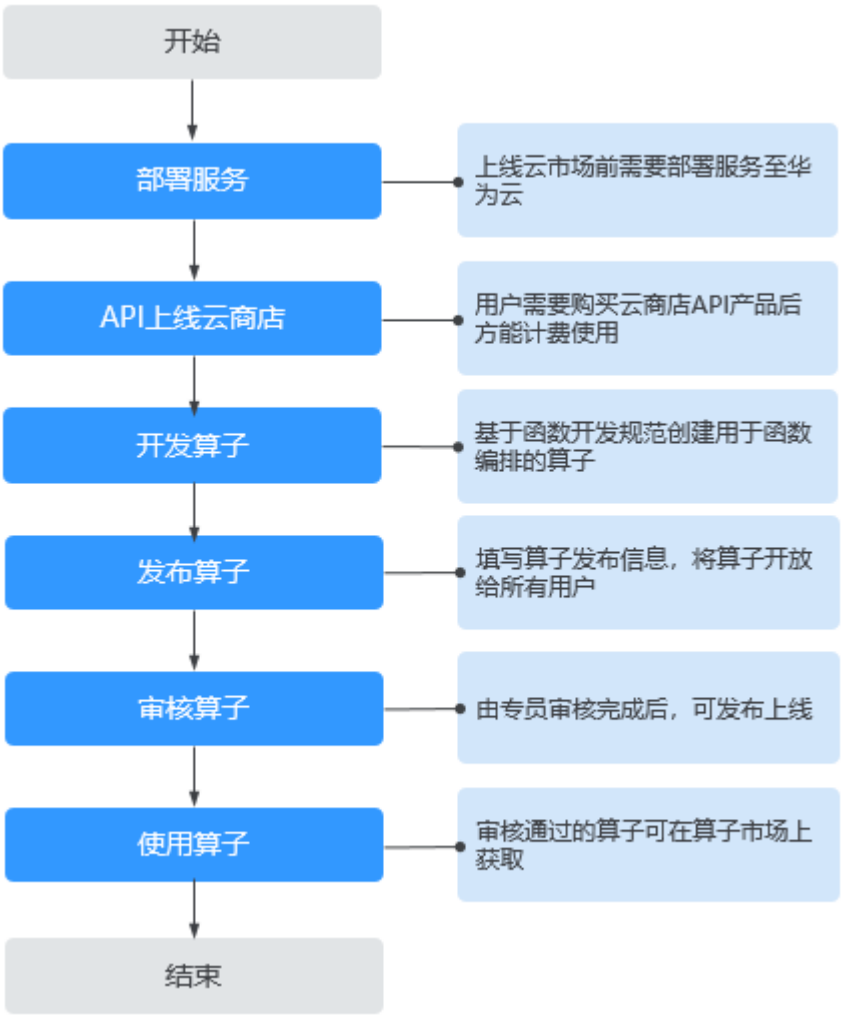
当算子被华为云用户使用之后，算子发布者将会获得相应的收益，详情参见**计费说明**。

算子从API上线、开发、到发布的整个流程如**图2-2**所示，本节主要介绍如何使用DWR发布已开发好的算子到算子市场。

说明

算子开发完成后，您可以参考**抽帧截图（自定义算子）**来验证算子。

图 2-2 算子发布流程



约束与限制

当前发布算子功能仅支持IAM主账号。

前提条件

算子已开发完成，开发规范参考**自定义函数开发规范**，建议使用Go、Python开发（算子的冷启动效果更好）。

算子已在华为云云商店上架，上架指导参考**发布API类商品操作指导**。

说明

开发的新算子如果要支持同步工作流，需要保证同步工作流最后一个算子返回方式为以下的一种：

- 返回方式1：字符串数据

```
{
   "execution_name":"84a3dd2bd67f43aa9b98cdd74604ca68", //工作流实例名称
   "graph_name":"test_workflow", //工作流名称
   "Records":[  // 处理对象

   ],
    "dynamic_source": {//执行算子的输出结果
    "tasks": [
       {body},  // 直接返回body字符串
   ]
  }
}
```

- 返回方式2：文件流数据

```
{
   "execution_name":"84a3dd2bd67f43aa9b98cdd74604ca68", //工作流实例名称
   "graph_name":"test_workflow", //工作流名称
   "Records":[  // 处理对象

   ],
   "dynamic_source":{ //执行算子的输出结果
   "tasks":[
         {
               "output":{  // 同步返回的输出文件地址：桶名、对象名、区域
               "bucket":"bucketname",
               "object":"objectname",
               "location":"cn-north-4"
            }
         }
      ]
   }
}
```

操作步骤

步骤1 登录管理控制台。在左侧导航栏上方，单击 ☰ ，选择“存储 > 数据工坊DWR”。进入DWR页面。

步骤2 在左侧导航栏选中“发布算子”，进入“发布算子”页面。

步骤3 单击界面右上角的“发布公共算子”。

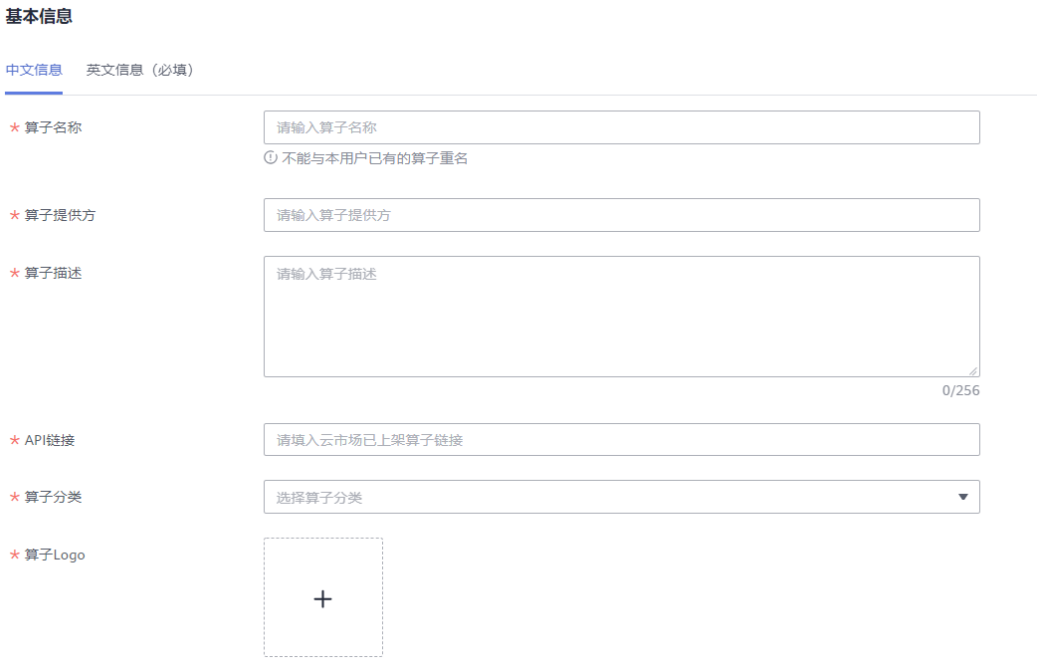
步骤4 配置算子基本信息。

表 2-1 配置基本信息

参数	说明
算子名称	算子名称不能与本用户已有的算子重名。
算子提供方	-
算子描述	-
API链接	填写华为云云商店已上架的算子链接。

参数	说明
算子分类	根据算子市场上提供的分类进行选择。
算子logo	支持主流图片格式。

图 2-3 算子基本信息



步骤5 上传算子文件。

上传文件包括：算子代码包、中文和英文帮助文档、中文和英文服务协议文档、中文和英文开源声明、中文和英文测试报告。

图 2-4 上传文件

步骤6 配置算子参数。

参数	说明
Inputs参数（可选）	提供算子入参中动态参数对应的参数值，其Key值需要与动态参数中保持一致。
动态参数（可选）	提供算子入参的参数列表，可配置静态参数或动态参数，动态参数需要从Inputs参数中根据定义的Key取值。
权限版本	华为云统一身份认证（IAM）的权限版本，1.0版本以服务为粒度，提供有限的服务相关角色用于授权。1.1版本支持细粒度授权，可以精确到具体服务的操作、资源以及请求条件等。

步骤7 单击右下角的“提交审核”。

审核通过后，算子将发布至算子市场。您可以过滤第三方的算子提供方，查看您发布的算子。

----结束

算子参数配置示例

Inputs参数

Input结构体参数说明参见**创建工作流API**。

说明

regex参数设置的正则表达式请使用**regexploit**工具校验。

```
[
  {    //算子所在工作流输入列表
    "parameter_name": "bucket",
    "parameter_value": "",
    "value_type": "",
    "default": "",
    "type": "string",
    "label": "Body",
    "constraints": {
        "regex": ".*"//正则表达式请使用regexploit工具校验
    },
    "invisible": false,
    "description": "doc destination bucket name"
  }
]
```

动态参数

```
{
  "bucket": {
    "get_input": "$.inputs.bucket"//该值需要跟inputs参数中的parameter_name取值保持一致
  }
}
```

权限版本

obs授权参见**对象相关授权项**和**桶相关授权**。

```
[
   { //1.1版本支持细粒度授权，可以精确到具体服务的操作、资源以及请求条件等
     "version": "1.1",
     "statement": [
```

```
      {
      //对IAM用户组授予OBS指定资源的指定操作权限
        "action": [
          "obs:bucket:HeadBucket",
          "obs:bucket:ListBucketMultipartUploads",
          "obs:object:AbortMultipartUpload",
          "obs:object:PutObject",
          "obs:bucket:GetBucketAcl",
          "obs:object:GetObject"
        ]
      }
    ]
  }
]
```

2.3 官方算子一览

本小节介绍华为云自有服务提供的各类算子的参数配置说明。

说明

不同区域支持的算子可能不同，请以控制台实际为准。

表 2-2 华为云官方算子一览

分类	模板名称	是否支持同步工作流	算子提供方
图像处理	图像标签	是	图像识别服务 Image
	人脸检测	是	人脸识别服务 FRS
	人脸搜索	是	人脸识别服务 FRS
	人脸比对	是	人脸识别服务 FRS
	圆角剪切	是	对象存储服务 OBS
	普通裁剪	是	对象存储服务 OBS
	索引剪切	是	对象存储服务 OBS
	图片水印	是	对象存储服务 OBS
	文字水印	是	对象存储服务 OBS
	缩略图	是	对象存储服务 OBS

分类	模板名称	是否支持同步工作流	算子提供方
	内切圆裁剪	是	对象存储服务 OBS
	自适应旋转	是	对象存储服务 OBS
	基础旋转	是	对象存储服务 OBS
	质量变换	是	对象存储服务 OBS
	格式转换	是	对象存储服务 OBS
	图片暗水印	是	数据安全中心 DSC
视频处理	视频解析	是	媒体处理服务 MPC
	抽帧截图	是	媒体处理服务 MPC
	媒资转码	否	媒体处理服务 MPC
消息通知	DIS消息通知	NA	数据接入服务 DIS
	SMN消息通知	NA	消息通知服务 SMN
其他	事件延迟	NA	NA
	自定义	NA	NA

人脸比对

模板作用：人脸比对是将两个人脸进行比对，来判断是否为同一个人，返回比对置信度。如果传入的图片中包含多个人脸，选取最大的人脸进行比对。该模板实际调用的是FRS服务的人脸比对接口。

表 2-3 人脸比对属性配置说明

属性类别	参数名称	参数说明
基本属性	名称	任务的名称，修改后将体现在工作流编排区域。 ● 必须以字母或数字开头 ● 只能由字母、数字、下划线和中划线组成 ● 长度范围为1~20个字符 ● 不能和同一工作流中的其他任务重名
	超时(秒)	任务超时时间，即任务执行的最长时间。 支持设置0~300秒的超时时间，如果设置为0，则表示超时时间为默认值30秒。
	算子提供方	函数模板的提供方。
	错误处理	可定义不同类型错误发生时的重试次数、重试间隔，以及重试失败后跳转到的目标任务。 错误类型包括：匹配所有、执行失败、权限不合法、参数不合法、函数不存在、请求太频繁、函数不可用、函数异常
动态参数	bucket	待对比的人脸图片存放的OBS桶名。
	path	待对比的人脸图片在OBS桶中存放的目录。

人脸检测

模板作用：对输入图片进行人脸检测和分析，输出人脸在图像中的位置、人脸关键点位置和人脸关键属性。该模板实际调用的是FRS服务的人脸检测接口。

表 2-4 人脸检测属性配置说明

属性类别	参数名称	参数说明
基本属性	名称	任务的名称，修改后将体现在工作流编排区域。 ● 必须以字母或数字开头 ● 只能由字母、数字、下划线和中划线组成 ● 长度范围为1~20个字符 ● 不能和同一工作流中的其他任务重名
	超时(秒)	任务超时时间，即任务执行的最长时间。 支持设置0~300秒的超时时间，如果设置为0，则表示超时时间为默认值30秒。
	算子提供方	函数模板的提供方。

属性类别	参数名称	参数说明
	错误处理	可定义不同类型错误发生时的重试次数、重试间隔，以及重试失败后跳转到的目标任务。 错误类型包括：匹配所有、执行失败、权限不合法、参数不合法、函数不存在、请求太频繁、函数不可用、函数异常
动态参数	attributes	选择希望获取的属性列表。

人脸搜索

模板作用：在已有的人脸库中，查询与目标人脸相似的一张或者多张人脸，并返回相应的置信度。该模板实际调用的是FRS服务的**人脸搜索接口**。

表 2-5 人脸搜索属性配置说明

属性类别	参数名称	参数说明
基本属性	名称	任务的名称，修改后将体现在工作流编排区域。 • 必须以字母或数字开头 • 只能由字母、数字、下划线和中划线组成 • 长度范围为1~20个字符 • 不能和同一工作流中的其他任务重名
	超时(秒)	任务超时时间，即任务执行的最长时间。 支持设置0~300秒的超时时间，如果设置为0，则表示超时时间为默认值30秒。
	算子提供方	函数模板的提供方。
	错误处理	可定义不同类型错误发生时的重试次数、重试间隔，以及重试失败后跳转到的目标任务。 错误类型包括：匹配所有、执行失败、权限不合法、参数不合法、函数不存在、请求太频繁、函数不可用、函数异常
动态参数	face_set_name	人脸库名称。
	return_fields	指定返回的自定义字段。 数组类型参数必须以前括号([)开头，以后括号(])结尾，值之间用英文逗号(,)分隔。
	top_n	返回查询到的最相似的N张人脸，N默认为10。
	threshold	人脸相似度阈值，低于这个阈值则不返回，取值范围0~1，一般情况下建议取值0.93，默认为0。

圆角剪切

模板作用：指定圆角大小将图片剪切为圆角矩形。支持通过圆角半径大小和水平垂直大小两种方式设置。

表 2-6 圆角剪切参数

属性类别	参数名称	参数说明
基本属性	名称	任务的名称，修改后将体现在工作流编排区域。 ● 必须以字母或数字开头 ● 只能由字母、数字、下划线和中划线组成 ● 长度范围为1~20个字符 ● 不能和同一工作流中的其他任务重名
	超时(秒)	任务超时时间，即任务执行的最长时间。 支持设置0~300秒的超时时间，如果设置为0，则表示超时时间为默认值30秒。
	算子提供方	函数模板的提供方。
	错误处理	可定义不同类型错误发生时的重试次数、重试间隔，以及重试失败后跳转到的目标任务。 错误类型包括：匹配所有、执行失败、权限不合法、参数不合法、函数不存在、请求太频繁、函数不可用、函数异常
动态参数	crop_r	将图片切出圆角，指定圆角的半径，水平和垂直的值相同，可以使用像素数（如200）或百分比（如25p）。 像素数取值范围为[1，4096]，当像素数大于原图最小边的一半时，取最小边的二分之一。 百分比取值范围为[1p，50p]。 不能与rx和ry参数同时使用。
	crop_rx	圆角水平大小的参数，可以使用像素数（如200）或百分比（如25p）。 像素数取值范围为[1，4096]，当像素数大于原图最小边的一半时，取最小边的二分之一。 百分比取值范围为[1p，50p]。 需要与ry同时使用。
	crop_ry	圆角垂直大小的参数，可以使用像素数（如200）或百分比（如25p）。 像素数取值范围为[1，4096]，当像素数大于原图最小边的一半时，取最小边的二分之一。 百分比取值范围为[1p，50p]。 需要与rx同时使用。

属性类别	参数名称	参数说明
	crop_source_path	输出路径是否保持原路径。true，输出路径为：output/原路径；false，输出路径为：output。
	crop_outbucket	图片处理输出桶

普通裁剪

模板作用：可以设置图片上的任意一点为起始点，根据指定宽高进行图片剪切，剪切后的图片为矩形。

表 2-7 普通剪切

属性类别	参数名称	参数说明
基本属性	名称	任务的名称，修改后将体现在工作流编排区域。 ● 必须以字母或数字开头 ● 只能由字母、数字、下划线和中划线组成 ● 长度范围为1~20个字符 ● 不能和同一工作流中的其他任务重名
	超时(秒)	任务超时时间，即任务执行的最长时间。 支持设置0~300秒的超时时间，如果设置为0，则表示超时时间为默认值30秒。
	算子提供方	函数模板的提供方。
	错误处理	可定义不同类型错误发生时的重试次数、重试间隔，以及重试失败后跳转到的目标任务。 错误类型包括：匹配所有、执行失败、权限不合法、参数不合法、函数不存在、请求太频繁、函数不可用、函数异常
动态参数	crop_g	表示剪切起始原点位置。取值为tl、top、tr、left、center、right、bl、bottom和br，共9个取值。
	crop_h	剪切的高度，取值范围为[0，图片高度]。
	crop_w	剪切的宽度，取值范围为[0，图片宽度]。
	crop_x	表示剪切起始点的横坐标，默认左上角为原点。取值范围为[0，图片边界]。
	crop_y	表示剪切起始点的纵坐标，默认左上角为原点。取值范围为[0，图片边界]。

属性类别	参数名称	参数说明
	crop_source_path	输出路径是否保持原路径。true，输出路径为：output/原路径；false，输出路径为：output
	crop_outbucket	图片处理输出桶

索引剪切

模板作用：以图片左上角顶点为起始点，设宽为x轴，高为y轴。根据指定长度进行等长剪切，根据指定索引取出剪切后区域。

表 2-8 索引剪切

属性类别	参数名称	参数说明
基本属性	名称	任务的名称，修改后将体现在工作流编排区域。 ● 必须以字母或数字开头 ● 只能由字母、数字、下划线和中划线组成 ● 长度范围为1~20个字符 ● 不能和同一工作流中的其他任务重名
	超时(秒)	任务超时时间，即任务执行的最长时间。 支持设置0~300秒的超时时间，如果设置为0，则表示超时时间为默认值30秒。
	算子提供方	函数模板的提供方。
	错误处理	可定义不同类型错误发生时的重试次数、重试间隔，以及重试失败后跳转到的目标任务。 错误类型包括：匹配所有、执行失败、权限不合法、参数不合法、函数不存在、请求太频繁、函数不可用、函数异常
动态参数	crop_i	若剪切后一共有n块，则i的取值范围为[0，n-1]。0表示第一块，超过最大的块数则返回原图。
	crop_x	水平剪切的每块图片长度。取值范围为[1，图片宽度]。x和y参数只能任选其一。
	crop_y	垂直剪切的每块图片长度。取值范围为[1，图片高度]。 x和y参数只能任选其一。

属性类别	参数名称	参数说明
	crop_source_path	输出路径是否保持原路径。true，输出路径为：output/原路径；false，输出路径为：output
	crop_outbucket	图片处理输出桶

图片水印

模板作用：对添加水印的图片进行预处理操作，包括设置缩略、旋转图片和剪切图片，但不支持剪切为内切圆。

表 2-9 图片水印

属性类别	参数名称	参数说明
基本属性	名称	任务的名称，修改后将体现在工作流编排区域。 ● 必须以字母或数字开头 ● 只能由字母、数字、下划线和中划线组成 ● 长度范围为1~20个字符 ● 不能和同一工作流中的其他任务重名
	超时(秒)	任务超时时间，即任务执行的最长时间。 支持设置0~300秒的超时时间，如果设置为0，则表示超时时间为默认值30秒。
	算子提供方	函数模板的提供方。
	错误处理	可定义不同类型错误发生时的重试次数、重试间隔，以及重试失败后跳转到的目标任务。 错误类型包括：匹配所有、执行失败、权限不合法、参数不合法、函数不存在、请求太频繁、函数不可用、函数异常
动态参数	iwm_g	可选参数，表示水印处于图片的位置。取值为tl、top、tr、left、center、right、bl、bottom和br，共9个取值。默认值为tl。
	iwm_x	可选参数，表示距离图片边缘的水平距离，默认左上角为原点。取值范围为[0，4096]。默认值为10。单位为像素（px）。
	iwm_y	可选参数，表示距离图片边缘的垂直距离，默认左上角为原点。取值范围为[0，4096]。默认值为10。单位为像素（px）。

属性类别	参数名称	参数说明
	iwm_voffset	可选参数，表示水印距离图片水平中线的垂直偏移方向。可以使水印根据中线往上或往下偏移。取值范围为[-1000，1000]。默认值为0。单位为像素（px）。 当g取值为left、center、right才有意义，即位置为左中、正中、右中才有意义。
	iwm_align	可选参数，水印文字和图片的对齐方式。取值为0、1或2。默认值为0。 ● 0：表示上对齐。 ● 1：表示中对齐。 ● 2：表示下对齐。
	iwm_order	可选参数，水印文字和图片的前后顺序。取值为0或1。默认值为0。 ● 0：表示图片在前面。 ● 1：表示文字在前面。
	iwm_t	可选参数，文字或图片水印的透明度。取值范围为[0，100]。默认值为100，100%表示不透明。
	iwm_interval	可选参数，表示文字和图片的间距。取值范围为[0，1000]。
	iwm_image	水印图路径，添加图片水印时的必选参数。 图片水印地址为：*bucketName*/*objectName*（必须编码）或*bucketName*/*objectName*?x-image-process=*image*/*command*（必须编码） **须知** 内容必须是URL安全base64编码。 encodedObject = url_safe_base64_encode(object)。如object为“panda.png”，编码过后的内容为“cGFuZGEucG5n”。
	iwm_P	水印图片尺寸，大写的P，表示将水印图片按原图（指被添加水印的图片）比例百分比P进行缩放。取值范围为[1，100]。 **须知** 此处resize操作只支持大写P参数，不支持小写p参数。如需调整水印图片的大小，请参照**设置缩略**（除小p参数之外）。
	iwm_source_path	输出路径是否保持原路径。true，输出路径为：output/原路径；false，输出路径为：output
	iwm_outbucket	图片处理输出桶

文字水印

模板作用：对添加水印的图片进行预处理操作，包括设置缩略、旋转图片和剪切图片，但不支持剪切为内切圆。

表 2-10 文字水印

属性类别	参数名称	参数说明
基本属性	名称	任务的名称，修改后将体现在工作流编排区域。 ● 必须以字母或数字开头 ● 只能由字母、数字、下划线和中划线组成 ● 长度范围为1~20个字符 ● 不能和同一工作流中的其他任务重名
	超时(秒)	任务超时时间，即任务执行的最长时间。 支持设置0~300秒的超时时间，如果设置为0，则表示超时时间为默认值30秒。
	算子提供方	函数模板的提供方。
	错误处理	可定义不同类型错误发生时的重试次数、重试间隔，以及重试失败后跳转到的目标任务。 错误类型包括：匹配所有、执行失败、权限不合法、参数不合法、函数不存在、请求太频繁、函数不可用、函数异常
动态参数	twm_g	可选参数，表示水印处于图片的位置。取值为tl、top、tr、left、center、right、bl、bottom和br，共9个取值。默认值为tl。
	twm_x	可选参数，表示距离图片边缘的水平距离，默认左上角为原点。取值范围为[0，4096]。默认值为10。单位为像素（px）。
	twm_y	可选参数，表示距离图片边缘的垂直距离，默认左上角为原点。取值范围为[0，4096]。默认值为10。单位为像素（px）。
	twm_voffset	可选参数，表示水印距离图片水平中线的垂直偏移方向。可以使水印根据中线往上或往下偏移。取值范围为[-1000，1000]。默认值为0。单位为像素（px）。 当g取值为left、center、right才有意义，即位置为左中、正中、右中才有意义。
	twm_align	可选参数，水印文字和图片的对齐方式。取值为0、1或2。默认值为0。 ● 0：表示上对齐。 ● 1：表示中对齐。 ● 2：表示下对齐。
	twm_order	可选参数，水印文字和图片的前后顺序。取值为0或1。默认值为0。 ● 0：表示图片在前面。 ● 1：表示文字在前面。

属性类别	参数名称	参数说明
	twm_t	可选参数，文字或图片水印的透明度。取值范围为[0，100]。默认值为100，100%表示不透明。
	twm_interval	可选参数，表示文字和图片的间距。取值范围为[0，1000]。
	twm_text	添加文字水印时的必选参数。 **须知** 必须是URL安全base64编码。encodeText = url_safe_base64_encode(fontText)，最大长度为64个字符（支持最多16个中文字符）。
	twm_size	可选参数，表示文字水印的文字大小。取值范围为(0，1000]。默认值为40。
	twm_type	可选参数，表示文字水印的文字类型。默认值为wqy-zenhei（文泉驿正黑，编码后的值：d3F5LXplbmhlaQ）。 **须知** 必须是URL安全base64编码。encodeText = url_safe_base64_encode(fontType)。
	twm_color	可选参数，表示文字水印的文字颜色。 格式为六位十六进制颜色码，取值为000000到FFFFFF，默认值为黑色。
	twm_shadow	可选参数，表示文字水印的阴影透明度。取值范围为(0，100]。
	twm_fill	可选参数，表示水印的铺满效果。取值为0或1。 ● 0：无效果。 ● 1：铺满。
	twm_rotate	可选参数，表示文字水印的按顺时针旋转的角度。取值范围为(0，360)。
	twm_source_path	输出路径是否保持原路径。true，输出路径为：output/原路径；false，输出路径为：output。
	twm_outbucket	图片处理输出桶。

缩略图

模板作用：通过resize操作能够使图片按照一定规则进行缩放，支持按照指定宽高和比例进行缩放。

表 2-11 缩略图设置

属性类别	参数名称	参数说明
基本属性	名称	任务的名称，修改后将体现在工作流编排区域。 ● 必须以字母或数字开头 ● 只能由字母、数字、下划线和中划线组成 ● 长度范围为1~20个字符 ● 不能和同一工作流中的其他任务重名
	超时(秒)	任务超时时间，即任务执行的最长时间。 支持设置0~300秒的超时时间，如果设置为0，则表示超时时间为默认值30秒。
	算子提供方	函数模板的提供方。
	错误处理	可定义不同类型错误发生时的重试次数、重试间隔，以及重试失败后跳转到的目标任务。 错误类型包括：匹配所有、执行失败、权限不合法、参数不合法、函数不存在、请求太频繁、函数不可用、函数异常
动态参数	resize_m	设置缩略的类型。取值为lfit、mfit、fill、pad和fixed，默认值为lfit。 ● lfit：指定一个w和h的矩形，将图片进行等比缩放，取在矩形内最大的图片。 ● mfit：指定一个w和h的矩形，将图片进行等比缩放，取在矩形延伸区域的最小图片。 ● fill：指定一个w和h的矩形，将图片进行等比缩放，取在延伸区域的最小图片，并进行居中剪切。即将mfit缩略类型的图片进行居中剪裁。 ● pad：指定一个w和h的矩形，将图片进行等比缩放，取在矩形内最大的图片，并在矩形空白处进行颜色填充。即lfit缩略类型的图片在矩形空白处进行颜色填充。 ● fixed：强制按照固定的宽高进行缩略。

属性类别	参数名称	参数说明
	resize_p	等比例缩放的倍数百分比。使用参数p时，无法使用其它参数。取值范围为[1，1000]。当取值为： ● ＜100：缩小。 ● =100：保持原图大小。 ● ＞100：放大。
	resize_h	目标缩略图的高度。取值为[1，4096]。
	resize_w	目标缩略图的宽度。取值为[1，4096]。
	resize_l	指定目标缩略图的最长边。取值为[1，4096]。 长边为指定的值，短边按照比例缩放。
	resize_s	指定目标缩略图的最短边。取值为[1，4096]。 短边为指定的值，长边按照比例缩放。
	resize_color	填充的颜色。选择pad（缩略后填充）模式时可以使用。 格式为十六进制颜色码，取值为000000到FFFFFF，默认值为白色。
	resize_limit	是否在目标缩略图比原图大时进行限制放大。取值为0或1，默认值为1。 ● 0：不进行限制。 ● 1：进行限制。
	iwm_path	输出路径是否保持原路径。true，输出路径为：output/原路径；false，输出路径为：output
	iwm_bucket	图片处理输出桶

内切圆裁剪

模板作用：以图片的中心为圆心，根据指定的半径进行图片剪切，剪切后的图片为圆形。

表 2-12 内切圆裁剪

属性类别	参数名称	参数说明
基本属性	名称	任务的名称，修改后将体现在工作流编排区域。 ● 必须以字母或数字开头 ● 只能由字母、数字、下划线和中划线组成 ● 长度范围为1~20个字符 ● 不能和同一工作流中的其他任务重名
	超时(秒)	任务超时时间，即任务执行的最长时间。 支持设置0~300秒的超时时间，如果设置为0，则表示超时时间为默认值30秒。
	算子提供方	函数模板的提供方。
	错误处理	可定义不同类型错误发生时的重试次数、重试间隔，以及重试失败后跳转到的目标任务。 错误类型包括：匹配所有、执行失败、权限不合法、参数不合法、函数不存在、请求太频繁、函数不可用、函数异常
动态参数	crop_ir	图片剪切的圆形半径，取值范围为[0，图片最短边的一半]。
	crop_path	输出路径是否保持原路径。true，输出路径为：output/原路径；false，输出路径为：output
	crop_bucket	图片处理输出桶

自适应旋转

模板作用：设置自适应方向，带有方向参数的图片会先根据方向参数信息进行自动旋转。

表 2-13 自适应旋转

属性类别	参数名称	参数说明
基本属性	名称	任务的名称，修改后将体现在工作流编排区域。 ● 必须以字母或数字开头 ● 只能由字母、数字、下划线和中划线组成 ● 长度范围为1~20个字符 ● 不能和同一工作流中的其他任务重名
	超时(秒)	任务超时时间，即任务执行的最长时间。 支持设置0~300秒的超时时间，如果设置为0，则表示超时时间为默认值30秒。

属性类别	参数名称	参数说明
	算子提供方	函数模板的提供方。
	错误处理	可定义不同类型错误发生时的重试次数、重试间隔，以及重试失败后跳转到的目标任务。 错误类型包括：匹配所有、执行失败、权限不合法、参数不合法、函数不存在、请求太频繁、函数不可用、函数异常
动态参数	rotate_value	按照顺时针旋转的角度，取值范围为[0，360]。 默认值为0，0表示不旋转。数值越大，图片按顺时针方向旋转的角度越大。
	autorate_source_path	输出路径是否保持原路径。true，输出路径为：output/原路径；false，输出路径为：output。
	autoratote_outbucket	图片处理输出桶。
	autoratote_outpath	图片处理输出路径。

基础旋转

模板作用：对图片进行旋转设置后，图片将会按顺时针方向进行旋转。

表 2-14 基础旋转

属性类别	参数名称	参数说明
基本属性	名称	任务的名称，修改后将体现在工作流编排区域。 ● 必须以字母或数字开头 ● 只能由字母、数字、下划线和中划线组成 ● 长度范围为1~20个字符 ● 不能和同一工作流中的其他任务重名
	超时(秒)	任务超时时间，即任务执行的最长时间。 支持设置0~300秒的超时时间，如果设置为0，则表示超时时间为默认值30秒。
	算子提供方	函数模板的提供方。
	错误处理	可定义不同类型错误发生时的重试次数、重试间隔，以及重试失败后跳转到的目标任务。 错误类型包括：匹配所有、执行失败、权限不合法、参数不合法、函数不存在、请求太频繁、函数不可用、函数异常

属性类别	参数名称	参数说明
动态参数	rotate_value	按照顺时针旋转的角度，取值范围为[0，360]。 默认值为0，0表示不旋转。数值越大，图片按顺时针方向旋转的角度越大。
	rotate_source_path	输出路径是否保持原路径。true，输出路径为：output/原路径；false，输出路径为：output。
	rotate_outbucket	图片处理输出桶。
	rotate_outpath	图片处理输出路径。

质量变换

模板作用：可以对输出格式为jpg的图片进行图片压缩，不使用压缩则可能会使图片占用的空间变大。

表 2-15 质量变换

属性类别	参数名称	参数说明
基本属性	名称	任务的名称，修改后将体现在工作流编排区域。 ● 必须以字母或数字开头 ● 只能由字母、数字、下划线和中划线组成 ● 长度范围为1~20个字符 ● 不能和同一工作流中的其他任务重名
	超时(秒)	任务超时时间，即任务执行的最长时间。 支持设置0~300秒的超时时间，如果设置为0，则表示超时时间为默认值30秒。
	算子提供方	函数模板的提供方。
	错误处理	可定义不同类型错误发生时的重试次数、重试间隔，以及重试失败后跳转到的目标任务。 错误类型包括：匹配所有、执行失败、权限不合法、参数不合法、函数不存在、请求太频繁、函数不可用、函数异常
动态参数	quality_q	图片的相对质量，即将图片按照原图的q%进行压缩。取值范围为[1，100]。 压缩公式：目标图质量=原图质量 * q% 例如：如果原图质量为100%，将图片压缩至80%的相对质量，则目标图片的质量为80%。如果原图的质量为80%，将图片压缩至80%的相对质量，则目标图片的质量为64%。

属性类别	参数名称	参数说明
	quality_Q	图片的绝对质量，即直接将图片压缩为Q%，与原图不存在相对关系，不依赖于原图。取值范围为[1，100]。 压缩公式： ● 原图质量 > Q%，目标图质量 = Q% ● 原图质量 = Q%，目标图质量 = 原图质量 = Q% ● 原图质量 < Q%，目标图质量 = 原图质量 例如：如果原图质量为100%，将图片压缩至80%的绝对质量，则目标图片的质量为80%。如果原图质量为70%，将图片压缩至80%的绝对质量，则目标图片的质量为70%。
	quality_path	输出路径是否保持原路径。true，输出路径为：output/原路径；false，输出路径为：output
	quality_bucket	图片处理输出桶

格式转换

模板作用：可以将原图转换为支持的图片格式。

- 支持的原图格式：jpg、jpeg、png、bmp、webp、gif、tiff。
- 支持输出的目标图格式：jpg、png、bmp、webp。

表 2-16 格式转换

属性类别	参数名称	参数说明
基本属性	名称	任务的名称，修改后将体现在工作流编排区域。 ● 必须以字母或数字开头 ● 只能由字母、数字、下划线和中划线组成 ● 长度范围为1~20个字符 ● 不能和同一工作流中的其他任务重名
	超时(秒)	任务超时时间，即任务执行的最长时间。 支持设置0~300秒的超时时间，如果设置为0，则表示超时时间为默认值30秒。
	算子提供方	函数模板的提供方。

属性类别	参数名称	参数说明
	错误处理	可定义不同类型错误发生时的重试次数、重试间隔，以及重试失败后跳转到的目标任务。 错误类型包括：匹配所有、执行失败、权限不合法、参数不合法、函数不存在、请求太频繁、函数不可用、函数异常
动态参数	format	将原图转换为支持的图片格式
	format_path	输出路径是否保持原路径。true，输出路径为：output/原路径；false，输出路径为：output
	format_bucket	图片处理输出桶

图像标签

模板作用：能准确识别自然图片中数百种场景、上千种通用物体及其属性。让智能相册管理等功能更加直观。该模板实际调用的是Image服务的**图像标签接口**。

表 2-17 图像标签属性配置说明

属性类别	参数名称	参数说明
基本属性	名称	任务的名称，修改后将体现在工作流编排区域。 • 必须以字母或数字开头 • 只能由字母、数字、下划线和中划线组成 • 长度范围为1~20个字符 • 不能和同一工作流中的其他任务重名
	超时(秒)	任务超时时间，即任务执行的最长时间。 支持设置0~300秒的超时时间，如果设置为0，则表示超时时间为默认值30秒。
	算子提供方	函数模板的提供方。
	错误处理	可定义不同类型错误发生时的重试次数、重试间隔，以及重试失败后跳转到的目标任务。 错误类型包括：匹配所有、执行失败、权限不合法、参数不合法、函数不存在、请求太频繁、函数不可用、函数异常
动态参数	threshold	置信度的阈值（0~100），输入非该范围内值算法即取默认值。默认值：0。
	limit	最多返回的tag数，默认值：30。
	language	中文：返回标签的语言类型为中文。 英文：返回标签的语言类型为英文。

视频解析

模板作用：用于新建视频解析任务，以解析视频元数据。该模板实际调用的是MPC服务的**新建视频解析任务接口**。

表 2-18 视频解析属性配置说明

属性类别	参数名称	参数说明
基本属性	名称	任务的名称，修改后将体现在工作流编排区域。 ● 必须以字母或数字开头 ● 只能由字母、数字、下划线和中划线组成 ● 长度范围为1~20个字符 ● 不能和同一工作流中的其他任务重名
	超时(秒)	任务超时时间，即任务执行的最长时间。 支持设置0~300秒的超时时间，如果设置为0，则表示超时时间为默认值30秒。
	算子提供方	函数模板的提供方。
	错误处理	可定义不同类型错误发生时的重试次数、重试间隔，以及重试失败后跳转到的目标任务。 错误类型包括：匹配所有、执行失败、权限不合法、参数不合法、函数不存在、请求太频繁、函数不可用、函数异常
动态参数	sync	视频解析处理模式。 ● 排队处理：查询后仅返回任务ID，还需进一步调用**查询视频解析任务接口**才能获取到视频元数据。 ● 同步处理：查询后将直接返回视频元数据。
	bucket	视频解析输出桶，用于保存解析后的视频。 输出桶需要和DWR工作流在同一区域，工作流所属区域为创建工作流的桶所属区域。例如工作流A是在桶A中创建的，则桶A的区域即为工作流A的区域。
	outpath	视频解析输出桶中存放视频的具体目录。 例如：输入abc或abc/，均表示视频存放在abc文件夹下，如果文件夹不存在，会自动新建。输出路径为空表示存放在桶的根目录。

抽帧截图

模板作用：用于新建视频截图任务。该模板实际调用的是MPC服务的**新建截图任务接口**。

表 2-19 抽帧截图属性配置说明

属性类别	参数名称	参数说明
基本属性	名称	任务的名称，修改后将体现在工作流编排区域。 ● 必须以字母或数字开头 ● 只能由字母、数字、下划线和中划线组成 ● 长度范围为1~20个字符 ● 不能和同一工作流中的其他任务重名
	超时(秒)	任务超时时间，即任务执行的最长时间。 支持设置0~300秒的超时时间，如果设置为0，则表示超时时间为默认值30秒。
	算子提供方	函数模板的提供方。
	错误处理	可定义不同类型错误发生时的重试次数、重试间隔，以及重试失败后跳转到的目标任务。 错误类型包括：匹配所有、执行失败、权限不合法、参数不合法、函数不存在、请求太频繁、函数不可用、函数异常
动态参数	bucket	抽帧截图输出桶，用于保存视频截图。 输出桶需要和工作流在同一区域，工作流所属区域为创建工作流的桶所属区域。例如工作流A是在桶A中创建的，则桶A的区域即为工作流A的区域。
	output	抽帧截图输出桶中存放截图的具体目录。 例如：输入abc或abc/，均表示视频存放在abc文件夹下，如果文件夹不存在，会自动新建。输出路径为空表示存放在桶的根目录。
	tar	是否压缩抽帧图片生成tar包。
	sync	是否同步处理，同步处理是指不下载全部文件，快速定位到截图位置进行截图。
	type	采样类型，可选择如下类型： ● 根据视频时长的百分比间隔采样 ● 根据时间间隔采样截图 ● 指定时间点截图

属性类别	参数名称	参数说明
	output_filename	截图输出文件名。 ● 如果只抽一张图（即：按DOTS方式，指定1个时间点）则按该指定文件名输出图片。 ● 如果抽多张图（即：按DOTS方式指定多个时间点或按TIME间隔截图）则输出图片名在该指定文件名基础上再增加时间点（示例：output_filename_10.jpg）。 ● 如果指定了压缩抽帧图片生成tar包，则tar包按该指定文件名输出。
	format	截图文件格式。 目前支持的取值为：jpg格式
	width	截图图片宽度。 取值范围：(96,3840] 单位：px
	height	截图图片高度。 取值范围：(96,2160] 单位：px
	maxlen	截图最长边的尺寸。宽边尺寸按照该尺寸与原始视频像素等比缩放计算。 取值范围：[240,3840] 单位：像素 **说明** 该参数和thumb_samp_width/thumb_samp_height选择使用，以thumb_samp_width/thumb_samp_height优先，若thumb_samp_width/thumb_samp_height都不等于0，则图片尺寸按thumb_samp_width/thumb_samp_height得出；反之，则图片尺寸按thumb_samp_maxlen得出。
	ratio	截图纵横比。
	percent	根据视频时长百分比间隔采样时的百分比值。
	dots	指定时间截图时的时间点数组。 数组类型参数必须以前括号([)开头，以后括号(])结尾，值之间用英文逗号(,)分隔。
	time	采样截图的时间间隔值。 单位：秒
	start	采样类型为“TIME”模式的开始时间，和thumb_samp_time配合使用。 单位：秒

属性类别	参数名称	参数说明
	duration	采样类型为“TIME”模式的持续时间，和thumb_samp_time、thumb_samp_start配合使用。表示从视频文件的第“thumb_samp_start”开始，持续时间为“thumb_samp_duration”，每间隔“thumb_samp_time”生成一张截图。 取值范围：[数字，ToEND]。“ToEND”表示持续到视频结束。 单位：秒 **说明** “thumb_samp_duration”必须大于等0，若设置为0，则截图持续时间从“thumb_samp_start”到视频结束。

媒资转码

模板作用：执行MPC服务的转码任务对视频进行转码，并在转码过程中压制水印、内容质检、视频截图等。该模板实际调用的是MPC服务的**新建转码任务接口**。

表 2-20 媒资转码属性配置说明

属性类别	参数名称	参数说明
基本属性	名称	任务的名称，修改后将体现在工作流编排区域。 ● 必须以字母或数字开头 ● 只能由字母、数字、下划线和中划线组成 ● 长度范围为1~20个字符 ● 不能和同一工作流中的其他任务重名
	超时(秒)	任务超时时间，即任务执行的最长时间。 支持设置0~300秒的超时时间，如果设置为0，则表示超时时间为默认值30秒。
	算子提供方	函数模板的提供方。
	错误处理	可定义不同类型错误发生时的重试次数、重试间隔，以及重试失败后跳转到的目标任务。 错误类型包括：匹配所有、执行失败、权限不合法、参数不合法、函数不存在、请求太频繁、函数不可用、函数异常
动态参数（基本参数）	bucket	媒资转码输出桶，用于保存转码后的视频文件。 输出桶需要和工作流在同一区域，工作流所属区域为创建工作流的桶所属区域。例如工作流A是在桶A中创建的，则桶A的区域即为工作流A的区域。

属性类别	参数名称	参数说明
	output	媒资转码输出桶中存放转码后视频的具体目录。 例如：输入abc或abc/，均表示视频存放在abc文件夹下，如果文件夹不存在，会自动新建。输出路径为空表示存放在桶的根目录。
	trans_tid	转码模板ID，数组，每一路转码输出对应一个转码配置模板ID，最多支持9个模板ID。 数组类型参数必须以前括号([)开头，以后括号(])结尾，值之间用英文逗号(,)分隔。
动态参数（字幕参数）	subtitle_type	媒资转码字幕类型。
	bucket	存放字幕文件的OBS桶。
	file_name	字幕对象名，即字幕文件。
动态参数（图片水印设置）	bucket	存放水印图片的OBS桶。
	file_name	水印图片对象名，即水印文件。
	dx	水印图片起点相对输出视频顶点的水平偏移量。 设置方法有如下两种： ● 整数型：表示图片起点水平偏移视频顶点的像素值，单位px。取值范围：[0，4096] ● 小数型：表示图片起点相对于视频分辨率宽的水平偏移比率。取值范围：(0，1)，支持4位小数，如0.9999，超出部分系统自动丢弃。 示例：输出视频分辨率宽1920，设置“dx”为“0.1”，“referpos”为“TopRight”（右上角），则水印图片右上角到视频右顶点在水平方向上偏移距离为192。
	dy	水印图片起点相对输出视频顶点的垂直偏移量。 设置方法有如下两种： ● 整数型：表示图片起点垂直偏移视频顶点的像素值，单位px。取值范围：[0，4096] ● 小数型：表示图片起点相对于视频分辨率高的垂直偏移比率。取值范围：(0，1)，支持4位小数，如0.9999，超出部分系统自动丢弃。 示例：输出视频分辨率高1080，设置“dy”为“0.1”，“referpos”为“TopRight”（右上角），则水印图片右上角到视频右顶点在垂直方向上的偏移距离为108。

属性类别	参数名称	参数说明
	referpos	水印位置。支持如下位置： • 右上角 • 左上角 • 右下角 • 左下角
	timeline_start	水印开始时间，与trans_iwm_tduration配合使用。 取值范围：数字 单位：秒
	timeline_duration	水印持续时间，与“trans_iwm_tstart”配合使用。 取值范围：[数字，ToEND]。“ToEND”表示持续到视频结束。
	image_process	水印图片处理方式。支持如下方式： • 简单缩放 • 图片变灰 • 透明化
	width	水印图片宽度，值有两种形式： • 整数型代水印图片宽的像素值，范围[8，4096]，单位px。 • 小数型代表相对输出视频分辨率宽的比率，范围(0,1)，支持4位小数，如0.9999，超出部分系统自动丢弃。
	height	水印图片高度，值有两种形式： • 整数型代表水印图片高的像素值，范围[8，4096]，单位px。 • 小数型代表相对输出视频分辨率高的比率，范围(0，1)，支持4位小数，如0.9999，超出部分系统自动丢弃。
	base	水印叠加母体，取值如下： • 叠加在输入片源 • 叠加在转码输出文件
动态参数（水印参数）	template_id	水印模板ID。可通过**新建水印模板**接口创建水印模板。

属性类别	参数名称	参数说明
动态参数（文字水印配置）	text_context	文字水印内容，内容需做Base64编码，此配置项不能为空 示例：若想添加文字水印“测试文字水印”，那么Content的值为：5rWL6K+V5paH5a2X5rC05Y2w
	dx	文字水印起点相对输出视频顶点的水平偏移量。 设置方法有如下两种： ● 整数型：表示文字起点水平偏移视频顶点的像素值，单位px。取值范围：[0，4096] ● 小数型：表示文字起点相对于视频分辨率宽的水平偏移比率。取值范围：(0，1)，支持4位小数，如0.9999，超出部分系统自动丢弃。 示例：输出视频分辨率宽1920，设置“dx”为“0.1”，“referpos”为“TopRight”（右上角），则文字水印右上角到视频右顶点在水平方向上偏移距离为192。
	dy	文字水印起点相对输出视频顶点的垂直偏移量。 设置方法有如下两种： ● 整数型：表示文字起点垂直偏移视频顶点的像素值，单位px。取值范围：[0，4096] ● 小数型：表示文字起点相对于视频分辨率高的垂直偏移比率。取值范围：(0，1)，支持4位小数，如0.9999，超出部分系统自动丢弃。 示例：输出视频分辨率高1080，设置“dy”为“0.1”，“referpos”为“TopRight”（右上角），则文字水印右上角到视频右顶点在垂直方向上的偏移距离为108。
	referpos	文字水印位置。支持如下位置： ● 右上角 ● 左上角 ● 右下角 ● 左下角
	timeline_start	文字水印开始时间，与“trans_twm_tduration”配合使用。 取值范围：数字 单位：秒
	timeline_duration	文字水印持续时间，与“trans_twm_tstart”配合使用。 取值范围：[数字，ToEND]。“ToEND”表示持续到视频结束。

属性类别	参数名称	参数说明
	base	文字水印叠加母体，取值如下： ● 叠加在输入片源 ● 叠加在转码输出文件
	font_name	文字水印文字字体名。当前支持的字体有 ● 方正黝黑 ● 微软雅黑
	font_size	文字水印文字字体大小。 取值范围：[4, 120]
	font_color	文字水印文字字体颜色。

DIS 消息通知

模板作用：发送消息到您指定的DIS通道。该模板实际调用的是DIS服务的**上传数据接口**。

表 2-21 DIS 消息通知属性配置说明

属性类别	参数名称	参数说明
基本属性	名称	任务的名称，修改后将体现在工作流编排区域。 ● 必须以字母或数字开头 ● 只能由字母、数字、下划线和中划线组成 ● 长度范围为1~20个字符 ● 不能和同一工作流中的其他任务重名
	超时(秒)	任务超时时间，即任务执行的最长时间。 支持设置0~300秒的超时时间，如果设置为0，则表示超时时间为默认值30秒。
	算子提供方	函数模板的提供方。
	错误处理	可定义不同类型错误发生时的重试次数、重试间隔，以及重试失败后跳转到的目标任务。 错误类型包括：匹配所有、执行失败、权限不合法、参数不合法、函数不存在、请求太频繁、函数不可用、函数异常
动态参数	stream_name	已创建的DIS通道名称。
	partition_id	DIS通道分区的唯一标识符。
	partition_key	数据将写入的DIS通道分区。

SMN 消息通知

模板作用：可用于在执行某项任务后，向SMN主题的订阅者发送通知。您可以在工作流的任意位置添加SMN消息通知，将上一个函数的执行结果发送给订阅者。

表 2-22 SMN 消息通知属性配置说明

属性类别	参数名称	参数说明
基本属性	名称	任务的名称，修改后将体现在工作流编排区域。 ● 必须以字母或数字开头 ● 只能由字母、数字、下划线和中划线组成 ● 长度范围为1~20个字符 ● 不能和同一工作流中的其他任务重名
	超时(秒)	任务超时时间，即任务执行的最长时间。 支持设置0~300秒的超时时间，如果设置为0，则表示超时时间为默认值30秒。
	算子提供方	函数模板的提供方。
	错误处理	可定义不同类型错误发生时的重试次数、重试间隔，以及重试失败后跳转到的目标任务。 错误类型包括：匹配所有、执行失败、权限不合法、参数不合法、函数不存在、请求太频繁、函数不可用、函数异常
动态参数	topic_urn	SMN topic唯一标识。选择已授权给OBS发布消息的SMN主题，以便向主题订阅者发送消息。SMN主题需通过SMN页面创建。 SMN服务的操作指导请参见《消息通知服务用户指南》中“**创建主题**”、“**设置主题策略**”和“**订阅主题**”章节的内容。 **说明** SMN主题配置成功后，请不要随意删除与OBS DWR工作流相关联的主题，也不要取消主题对OBS的授权。若与OBS DWR工作流相关联的主题被删除或取消该主题对OBS的授权，可能会导致对应主题的订阅者无法收到消息。 下拉列表中仅展示与DWR工作流同区域且同项目的SMN主题。工作流所属区域为创建工作流的桶所属区域。例如工作流A是在桶A中创建的，则桶A的区域即为工作流A的区域。
	project_id	SMN topic主题名称，为发布消息的标题，给邮箱订阅者发送邮件时作为邮件主题。

事件延迟

模板作用：可用于控制工作流两个相邻任务间的等待时长，例如执行任务A后，规定等待一段时间再继续执行任务B。

表 2-23 事件延迟属性配置说明

属性类别	参数名称	参数说明
基本属性	名称	任务的名称，修改后将体现在工作流编排区域。 ● 必须以字母或数字开头 ● 只能由字母、数字、下划线和中划线组成 ● 长度范围为1~20个字符 ● 不能和同一工作流中的其他任务重名
	超时(秒)	执行下一任务前的等待时长，单位：秒。支持设置的范围为1~86400秒。

自定义

用户可自定义函数，满足不同场景的任务定制需求。

自定义函数属性配置说明见下表，另外需要遵循**自定义函数开发规范**。

表 2-24 自定义函数属性配置说明

属性类别	参数名称	参数说明
基本属性	名称	任务的名称，修改后将体现在工作流编排区域。 ● 必须以字母或数字开头 ● 只能由字母、数字、下划线和中划线组成 ● 长度范围为1~20个字符 ● 不能和同一工作流中的其他任务重名
	超时(秒)	任务超时时间，即任务执行的最长时间。 支持设置0~300秒的超时时间，如果设置为0，则表示超时时间为默认值30秒。
	函数唯一标识	选择需要执行的FunctionGraph中的函数。函数需通过FunctionGraph页面创建。 FunctionGraph服务的操作指导请参见《函数工作流用户指南》中“**创建并初始化函数**”章节的内容。 下拉列表中仅展示与DWR工作流同区域且同项目的函数。工作流所属区域为创建工作流的桶所属区域。例如工作流A是在桶A中创建的，则桶A的区域即为工作流A的区域。
动态参数		若自定义函数中存在动态参数，可以指定动态参数的参数名和取值，作为函数的输入。

图片暗水印

图片暗水印指将水印以不可见的形式添加到图片中，既保证了水印不会影响图片美观性，又保证了图片的原创性。当图片被盗用后，您可对图片进行暗水印解码，验证版权归属。

表 2-25 图片暗水印

属性类别	参数名称	参数说明
基本属性	名称	任务的名称，修改后将体现在工作流编排区域。 ● 必须以字母或数字开头 ● 只能由字母、数字、下划线和中划线组成 ● 长度范围为1~20个字符 ● 不能和同一工作流中的其他任务重名
	超时(秒)	任务超时时间，即任务执行的最长时间。 支持设置0~300秒的超时时间，如果设置为0，则表示超时时间为默认值30秒。
	算子提供方	函数模板的提供方。
	错误处理	可定义不同类型错误发生时的重试次数、重试间隔，以及重试失败后跳转到的目标任务。 错误类型包括：匹配所有、执行失败、权限不合法、参数不合法、函数不存在、请求太频繁、函数不可用、函数异常
动态参数	image_watermark_path	作为图片盲水印文件的OBS路径，格式为obs://{bucket}/{object}。 ● bucket为和当前函数处于相同区域的OBS桶名称。 ● object为对象全路径名，支持jpg和png格式。
	output_using_input	输出路径是否保持原路径。 ● true，输出路径为：output_path/原路径。 ● false，输出路径为：output_path。
	output_bucket	嵌入水印后的图片输出的OBS桶名称，指定的OBS桶需和当前函数在相同区域项目下。
	output_path	嵌入水印后的图片输出路径，与参数output_using_input结合使用。

3 数据处理

3.1 数据处理介绍

DWR 如何实现数据处理

DWR提供的近数据处理能力，可以对OBS内存储的数据，按照用户编排的工作流进行自动化处理（如解析、转码、截图等）。

DWR基于函数工作流FunctionGraph的函数能力，将复杂的业务处理逻辑编排为工作流，通过事件触发器或API驱动，自动化完成多项复杂的数据处理任务。DWR提供图形化界面，方便用户直观便捷的构建数据处理流程，同时提供了预置的算子和自定义函数能力，覆盖数据处理的各种场景。预置算子的详细介绍，请参见**官方算子一览**。用户在自行开发自定义函数时，函数的输入参数和输出参数需要遵守**自定义函数开发规范**。

DWR支持异步和同步两种方式启动工作流，其中同步方式支持直接返回数据：

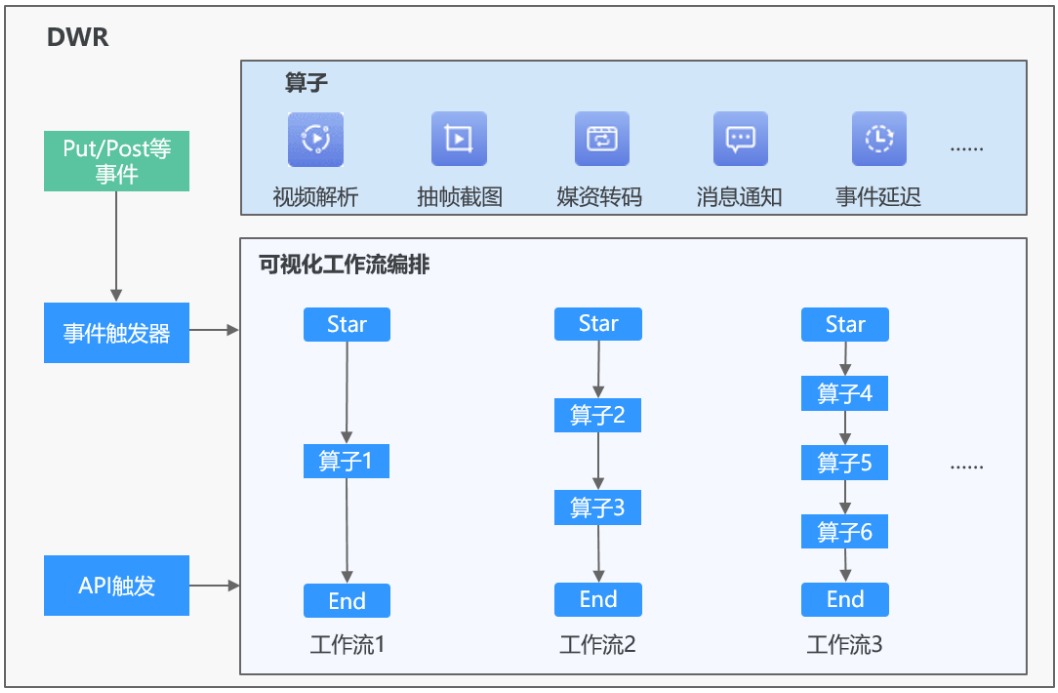
- 通过事件触发器启动工作流（异步方式）

  在OBS桶上配置事件触发器，指定工作流触发的条件，如桶内什么数据在执行某类操作后开始处理，当事件触发时异步执行满足条件的复杂任务。通常这类复杂任务处理逻辑相同，可以对一类对象进行操作。比较典型的场景是：用户上传视频对象后，可以根据工作流自动完成视频解析或者转码。

- 通过API启动工作流（同步和异步都支持）

  在少数场景下，用户对单个对象或者一类对象进行的复杂操作是有区别的，这就要求用户通过API调用方式来实现单个对象粒度的复杂任务处理，可以指定某个对象立即执行某个特定的工作流。

图 3-1 数据处理工作流



优势

- 简单易用：通过控制台的图形化界面，轻松按需搭建数据处理流程。
- 功能强大：支持华为云各种数据处理服务的工作流处理能力。
- 容错性好：通过内置错误重试能力，自动重试失败或超时的任务，对不同类型错误做出不同响应。同时提供工作流异常或失败后的恢复接口，从失败的位置继续执行工作流。

权限说明

请参见**权限管理**。

约束与限制

请参见**使用限制**。

使用方式

DWR支持通过控制台、API配置数据处理的工作流和事件触发器。

支持的使用方式	参考文档
控制台	• **创建工作流** • **创建事件触发器**
API	请参见**《数据工坊API参考》**。

前提条件

已创建工作流。

3.2 创建工作流

操作场景

工作流主要是对算子进行编排，这样DWR就可以按照用户编排的工作流对OBS中的数据进行自动化处理（如视频解析、图片转码、视频截图等）。

操作步骤

步骤1 登录管理控制台。在左侧导航栏上方，单击 ☰ ，选择“存储 > 数据工坊DWR”。

进入DWR页面。

步骤2 在左侧导航栏选中“工作流”，进入“工作流”页面。第一次进入时需要进行“统一授权”。

步骤3 单击界面右上角的“创建工作流”，进入“工作流编排”页面。

步骤4 将左侧预置的模板或自定义的函数拖拽至编排区域，同时在右侧属性面板配置基本属性和动态参数，配置完成后图标将由白色填充变为蓝色填充。

各预置模板及自定义函数的参数配置说明，请参见**官方算子一览**。

说明

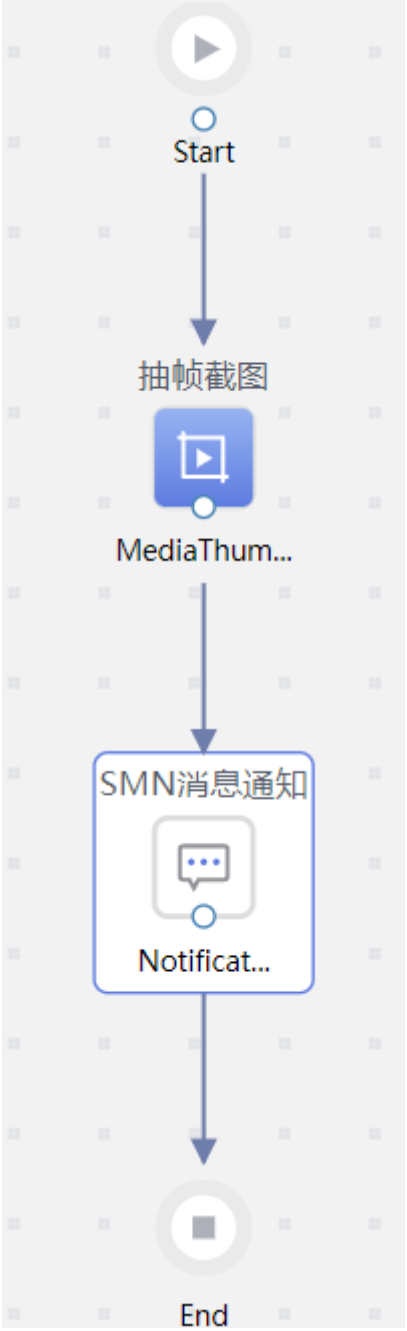
使用视频解析、抽帧截图、媒资转码等模板需要先在MPC中开启桶授权，详情请参见**权限说明**。

步骤5 鼠标单击各流程图标下方的小圆圈并长按拖拽，将工作流完整串联起来。

说明

当前仅支持串行工作流。

图 3-2 串联后的完整工作流



步骤6 单击右上角的“保存”，填写工作流基本信息，如表3-1所示。

创建完成的所有同区域工作流，都将在工作流列表展示。工作流创建完成后，还需要创建事件触发器，或通过API触发，工作流才能工作。

表 3-1 工作流基本信息

参数	说明
工作流名称	-
工作流类型	DWR支持同步和异步两种数据处理方式

参数	说明
匿名	对同步工作流设置是否可匿名访问

说明

- 同步工作流支持异步启动。
- 支持同步工作流的自营算子参见**官方算子一览**。
- 同步工作流最后一个算子当前仅支持以下两种返回方式：
  - 返回方式1：字符串数据

```
{
   "execution_name":"84a3dd2bd67f43aa9b98cdd74604ca68", //工作流实例名称
   "graph_name":"test_workflow", //工作流名称
   "Records":[  // 处理对象

   ],
    "dynamic_source": {//执行算子的输出结果
    "tasks": [
       {body},  // 直接返回body字符串
    ]
  }
}
```

  - 返回方式2：文件流数据

```
{
   "execution_name":"84a3dd2bd67f43aa9b98cdd74604ca68", //工作流实例名称
   "graph_name":"test_workflow", //工作流名称
   "Records":[  // 处理对象

   ],
   "dynamic_source":{ //执行算子的输出结果
   "tasks":[
         {
               "output":{  // 同步返回的输出文件地址：桶名、对象名、区域
               "bucket":"bucketname",
               "object":"objectname",
               "location":"cn-north-4"
            }
         }
      ]
   }
}
```

图 3-3 创建工作流

----结束

3.3 启动工作流

3.3.1 通过事件触发器异步启动工作流

操作场景

创建工作流之后，可以通过创建事件触发器来为工作流设置执行条件，指定桶内什么数据在执行某类操作后开始执行工作流。

前提条件

已完成工作流创建。

约束与限制

一个桶支持绑定10个事件触发器。

操作步骤

步骤1 登录管理控制台。在左侧导航栏上方，单击 ☰ ，选择“存储 > 数据工坊DWR”。

进入DWR页面。

步骤2 在左侧导航栏选择“工作流”，进入“工作流”页面。

步骤3 在工作流列表中，单击待关联工作流操作列的“创建事件触发器”进行触发器的创建。

图 3-4 创建事件触发器

步骤4 配置事件触发器参数。

表 3-2 事件触发器参数说明

参数	说明
触发器名称	事件触发器的名称，用户自定义。同一桶内的触发器名称不允许重复。
关联工作流	满足该事件触发器条件时，自动执行的工作流。
事件源存储桶	事件发生的源桶，即事件在该桶中发生时，触发关联工作流。 该参数不支持修改，默认为创建事件触发器的桶。

参数	说明
事件源类型	使事件触发器生效的事件源类型。目前，OBS支持以下事件源类型： • **ObjectCreated**：表示所有创建对象的操作，包含Put、Post、Copy对象以及合并段。 • **Put**：使用Put方法上传对象。 • **Post**：使用Post方法上传对象。 • **Copy**：使用Copy方法复制对象。 • **CompleteMultipartUpload**：表示合并分段任务。 • **ObjectRemoved**：表示删除对象。 • **Delete**：指定对象版本号删除对象。 • **DeleteMarkerCreated**：不指定对象版本号删除对象。 多个事件源类型可以作用于同一个目标对象，例如：同时选择“事件源类型”复选框中的**Put**、**Copy**、**Delete**等方法作用于某目标对象，则用户往该桶中上传、复制、删除符合前后缀规则的目标对象时，均会使触发器生效。**ObjectCreated**包含了**Put**、**Post**、**Copy**和**CompleteMultipartUpload**，如果选择了**ObjectCreated**，则不能再选择**Put**、**Post**、**Copy**或**CompleteMultipartUpload**。同理如果选择了**ObjectRemoved**，则不能再选择**Delete**或**DeleteMarkerCreated**。
前缀	使事件触发器生效的对象前缀。 **说明** 当前缀和后缀都不配置时，事件触发器将作用于桶中所有对象。
后缀	使事件触发器生效的对象后缀。 **说明** • 文件夹是以“/”结尾的，“/”前的字符为文件夹名称。若要对文件夹进行后缀匹配，后缀必须以“/”结尾。 • 当前缀和后缀都不配置时，事件触发器将作用于桶中所有对象。

步骤5 单击“确定”，完成事件触发器创建。

当事件触发器规则的条件满足时，将自动执行关联工作流定义的任务。

----结束

3.3.2 通过 API 异步启动工作流

DWR支持通过API异步启动已有工作流，详情参见**API异步启动工作流**。

3.3.3 通过 API 同步启动工作流

DWR支持通过API同步启动已有工作流。

请求示例

```
GET /objectkey?x-workflow-graph-name=gramname/p1_v1,p2_v2 HTTP/1.1
Host: bucket.obs.cn-north-4.myhuaweicloud.com
```

```
Authorization: OBS H4IPJX0TQTHTHEBQQCEC:sc2PM13Wlfcoc/YZLK0MwsI2Zpo=
Date: Thu, 27 Aug 2020 12:38:10 GMT

body(stream body/json body)
```

表 3-3 请求参数说明

参数	是否必选	参数类型	描述
x-workflow-graph-name	是	字符串	工作流名称和运行参数，比如：x-workflow-graph-name=gramname/p1_v1,p2_v2，graphname是工作流名称，p1_v1表示工作流运行参数p1对应的值为v1，p2对应的值为v2。 **注意** 当参数名和值中包含下划线时，需将下划线转义为“%5F”。 如：工作流名称为test-workflow，参数名p1为thumb_samp_maxlen，p1的值v1为500，参数名p2为thumb_samp_dots，p2的值v2为11，则请求为：<对象url>?x-workflow-graph-name=test-workflow/thumb**%5F**samp**%5F**maxlen_500,thumb**%5F**samp**%5F**dots_11

4 相关参考

4.1 自定义函数开发规范

用户在自行开发自定义函数时，函数的输入参数和输出参数需要遵守本节的开发规范。

自定义函数的编译方式请参考FunctionGraph的《开发指南》。

函数输入参数

工作流执行自定义函数时，函数输入参数的JSON格式的结构体和环境变量的定义如下：

表 4-1 函数输入的 JSON 格式体

名称	是否必选	参数类型	说明
execution_name	是	String	工作流实例名称。
graph_name	是	String	工作流名称。
Records	是	Array	工作流触发的事件源事件消息。
inputs	否	Map[String]String	用户可修改参数列表，可以为空。
dynamic_source	否	Map	函数执行必须的参数，可用于传入调用的服务。

表 4-2 函数的环境变量

名称	是否必选	参数类型	说明
region	否	String	当前区域名称。

函数输入的JSON示例

```
{
  "execution_name": "84a3dd2bd67f43aa9b98cdd74604ca68",//工作流实例名称
  "graph_name": "test_workflow",//工作流名称
  "Records": [
    {
      "eventName": "ObjectCreated:Put",//触发事件通知的事件名
      "eventRegion": "cn-north-4",//事件所在的region
      "eventSource": "OBS",//消息源，固定为"OBS"
      "eventTime": "2021-12-23T14:50:22.957Z",//事件时间，格式为ISO-8601，示例:
2020-07-10T09:24:11.418Z
      "eventVersion": "3.0",//版本号，目前为"3.0"
      "obs": {
        "Version": "1.0",
        "bucket": {
          "bucket": "examplebucket",//桶名
          "name": "examplebucket",//桶名
          "ownerIdentity": {
            "ID": "08b4efe0fc00d3ce0f17c01b948f6e80"//桶拥有者的账号ID
          }
        },
        "configurationId":"test-trigger",//此事件匹配的OBS中事件触发器的名称
        "object": {
          "eTag": "fc85a07cff68977bf5b2108e7436ca2d",//对象的etag
          "key": "exampleobject.docx",//对象名
          "oldpsxpth": "",//文件在并行文件系统中rename前的路径
          "sequencer": "1",//确定某个特定对象事件顺序的标识
          "size": "524298",//对象的大小
          "versionId": "G001017DE60E176D0000401106696610null"//对象的版本ID
        }
      },
      "requestParameters": {
        "sourceIPAddress": "x.x.x.x"//请求的源IP
      },
      "responseElements": {
        "x-obs-id-2": "",//帮助定位问题的特殊符号
        "x-obs-request-id": "84a3dd2bd67f43aa9b98cdd74604ca68"//请求对应的requestid
      },
      "userIdentity": {
        "ID": "08b4efe0fc00d3ce0f17c01b948f6e80"//触发事件的用户对应的计费ID
      }
    }
  ],
  "inputs": {  //执行工作流的输入参数
    "parametername": "parametervalue",
    "parametername": "parametervalue"
    },
  "dynamic_source": { //执行自定义函数的输入参数
    "parametername": "parametervalue",
    "parametername": "parametervalue"
  }
}
```

函数输出参数

函数输出参数的JSON格式的结构体定义如下:

表 4-3 函数输出的 JSON 格式体

名称	是否必选	参数类型	说明	约束
execution_name	是	String	工作流实例名称。	继承函数输入参数的execution_name。
graph_name	是	String	工作流名称。	继承函数输入参数的graph_name。
Records	是	Array	工作流触发的事件源事件消息。	如果没有变化，则继承函数输入参数的records。
inputs	否	Map[String]String	用户可修改参数列表。	如果没有新增，则继承函数输入参数的inputs。
dynamic_source	否	Map	函数的输出参数，可用于传递给下一个执行的函数。	-
operation_name	否	String	函数操作名。	系统内置的工作流函数操作名有： ● 视频解析：MPC.Metadata ● 视频截图：MPC.Thumbnail ● 视频转码：MPC.Transcode ● SMN消息通知：SMN.Publish

对接截图函数示例（GO 语言）

```
package main

import (
    "encoding/json"
    "errors"
    "go-runtime/go-api/context"
)

func DemoHandler(jsonData []byte, ctx context.RuntimeContext) (interface{}, error) {
    var eventMsg Payload
    err := json.Unmarshal(jsonData, &eventMsg)
    if err != nil {
        return nil, errors.New("not correct format")
    }
    // 存储输入桶和对象值
    record := eventMsg.Records[0]

    // 定义输出
    resp := struct {
        OBSMessages
        Inputs        map[string]interface{} `json:"inputs"`
        ExecutionName string                 `json:"execution_name"`
        GraphName     string                 `json:"graph_name"`
```



```
    DynamicSource struct {
        *CreateThumbnailDynamicSourceBody
    } `json:"dynamic_source"`
}{}
// 配置截图参数，为下游截图任务提供参数配置
resp.DynamicSource.CreateThumbnailDynamicSourceBody = &CreateThumbnailDynamicSourceBody{
    Thumbnails: []*ThumbnailCreateTaskBody{
        &ThumbnailCreateTaskBody{
            //源文件地址。
            Input: &FileAddr{
                Location:   "cn-north-1",
                BucketName: record.Obs.Bucket.Name,
                Object:     record.Obs.Object.Key,
            },
            //输出地址。
            Output: &FileAddr{
                Location:   "cn-north-1",
                BucketName: record.Obs.Bucket.Name,
                Object:     "thumb_out",
            },
            //是否压缩抽帧图片生成tar包。
            Tar: 0,
            //是否同步处理，同步处理是指不下载全部文件，快速定位到截图位置进行截图。
            Mode: 0,
            //截图参数
            ThumbnailParam: &ThumbnailParam{
                Type:           "DOTS",
                MaxLength:      0,
                Dots:           []int64{2, 10, 14}, // 截图的位置(s)
                OutputFileName: "default_cover.jpg",
            },
        },
    },
}
// 以下参数需要继承传递，方便工作流下游函数获取对应参数值
resp.Inputs = eventMsg.Inputs
resp.Records = eventMsg.Records
resp.GraphName = eventMsg.GraphName
resp.ExecutionName = eventMsg.ExecutionName
return resp, nil
}
```

对接转码函数示例（GO 语言）

```
package main

import (
    "encoding/json"
    "errors"
    "go-runtime/go-api/context"
)

func DemoTranscodeHandler(jsonData []byte, ctx context.RuntimeContext) (interface{}, error) {
    var eventMsg Payload
    err := json.Unmarshal(jsonData, &eventMsg)
    if err != nil {
        return nil, errors.New("not correct format")
    }
    // 存储输入桶和对象值
    record := eventMsg.Records[0]

    // 定义输出
    resp := struct {
        OBSMessages
        Inputs        map[string]interface{} `json:"inputs"`
        ExecutionName string                 `json:"execution_name"`
        GraphName     string                 `json:"graph_name"`
        DynamicSource struct {
            *CreateTranscodeDynamicSourceBody
```



```
    } `json:"dynamic_source"`
}{}
// 配置截图参数，为下游截图任务提供参数配置
resp.DynamicSource.CreateTranscodeDynamicSourceBody = &CreateTranscodeDynamicSourceBody{
    Transcodes: []*CreateTranscodeTaskBody{
        &CreateTranscodeTaskBody{
            //源文件地址。
            Input: &FileAddr{
                Location:   "cn-north-4",
                BucketName: record.Obs.Bucket.Name,
                Object:     record.Obs.Object.Key,
            },
            //输出地址。
            Output: &FileAddr{
                Location:   "cn-north-4",
                BucketName: record.Obs.Bucket.Name,
                Object:     "transcode_out",
            },
            TransTemplateID: []int{7000523, 7000524, 7000526, 7000528, 7000530, 7000538},
            OutputFilenames: []string{"out_file1", "out_file2", "out_file3", "out_file4", "out_file5", "out_file6"},
        },
    },
}
// 以下参数需要继承传递，方便工作流下游函数获取对应参数值
resp.Inputs = eventMsg.Inputs
resp.Records = eventMsg.Records
resp.GraphName = eventMsg.GraphName
resp.ExecutionName = eventMsg.ExecutionName
return resp, nil
}
```

结构体示例（GO 语言）

```
package main

type CreateTranscodeDynamicSourceBody struct {
    Transcodes []*CreateTranscodeTaskBody `json:"transcodes"`
}

type CreateTranscodeTaskBody struct {
    //源文件存储地址。
    Input *FileAddr `json:"input,omitempty"`
    //转码后的视频文件存储地址。
    Output *FileAddr `json:"output"`
    //转码模板ID，数组
    TransTemplateID []int `json:"trans_template_id,omitempty"`
    //支持图片水印和文字水印，最多支持20个。
    Watermarks []*Watermark `json:"watermarks,omitempty"`
    //任务优先级。
    Priority string `json:"priority,omitempty"`
    //输出文件名称，每一路转码输出对应一个名称，需要与转码模板ID数组的顺序对应。
    OutputFilenames []string `json:"output_filenames,omitempty"`
}
type Watermark struct {
    Input          *FileAddr       `json:"input,omitempty"`
    TemplateID     int             `json:"template_id,omitempty"`
    TextContext    string          `json:"text_context,omitempty"`
    ImageWatermark *ImageWatermark `json:"image_watermark,omitempty"`
    TextWatermark  *TextWatermark  `json:"text_watermark,omitempty"`
}
type TextWatermark struct {
    Dx               string `json:"dx,omitempty"`
    Dy               string `json:"dy,omitempty"`
    ReferPos         string `json:"referpos,omitempty"`
    TimelineStart    string `json:"timeline_start,omitempty"`
    TimelineDuration string `json:"timeline_duration,omitempty"`
    FontName         string `json:"font_name,omitempty"`
    FontSize         string `json:"font_size,omitempty"`
    FontColor        string `json:"font_color,omitempty"`
```

```
    Base              string `json:"base,omitempty"`
}
type ImageWatermark struct {
    Dx                string `json:"dx,omitempty"`
    Dy                string `json:"dy,omitempty"`
    ReferPos          string `json:"referpos,omitempty"`
    TimelineStart     string `json:"timeline_start,omitempty"`
    TimelineDuration string `json:"timeline_duration,omitempty"`
    ImageProcess      string `json:"image_process,omitempty"`
    Width             string `json:"width,omitempty"`
    Height            string `json:"height,omitempty"`
    Base              string `json:"base,omitempty"`
}

type CreateThumbnailDynamicSourceBody struct {
    Thumbnails []*ThumbnailCreateTaskBody `json:"thumbnails"`
}

//FileAddr 文件路径结构定义
type FileAddr struct {
    Location   string `json:"location"`
    BucketName string `json:"bucket"`
    Object     string `json:"object"`
}

type ThumbnailCreateTaskBody struct {
    //源文件地址。
    Input *FileAddr `json:"input"`
    //输出地址。
    Output *FileAddr `json:"output"`
    //是否压缩抽帧图片生成tar包。
    Tar int `json:"tar,omitempty"`
    //是否同步处理，同步处理是指不下载全部文件，快速定位到截图位置进行截图。
    Mode int `json:"sync,omitempty"`
    //截图参数
    ThumbnailParam *ThumbnailParam `json:"thumbnail_para"`
}

type ThumbnailParam struct {
    Type           string  `json:"type"`
    Time           int64   `json:"time,omitempty"`
    StartTime      int64   `json:"start_time,omitempty"`
    Duration       int64   `json:"duration,omitempty"`
    Dots           []int64 `json:"dots,omitempty"`
    Format         int64   `json:"format,omitempty"`
    AspectRatio    int64   `json:"aspect_ratio,omitempty"`
    Width          int64   `json:"width,omitempty"`
    Height         int64   `json:"height,omitempty"`
    MaxLength      int64   `json:"max_length,omitempty"`
    OutputFileName string  `json:"output_filename,omitempty"`
}

type OBSMessages struct {
    Records []OBSRecord `json:"Records"`
}

// OBSRecord OBS消息格式
type OBSRecord struct {
    EventVersion       string            `json:"eventVersion"`
    EventSource        string            `json:"eventSource"`
    EventRegion        string            `json:"eventRegion"`
    EventTime          string            `json:"eventTime"`
    EventName          string            `json:"eventName"`
    UserIdentity       UserIdentity      `json:"userIdentity"`
    RequestParameters RequestParameters `json:"requestParameters"`
    ResponseElements   ResponseElements  `json:"responseElements"`
    Obs                *OBSInfo          `json:"obs"`
}
```

```
// UserIdentity 用户id
type UserIdentity struct {
    ID string `json:"ID,omitempty"`
}

//RequestParameters 原始请求参数
type RequestParameters struct {
    SourceIPAddress string `json:"sourceIPAddress,omitempty"`
}

//ResponseElements 响应参数
type ResponseElements struct {
    OBSRequestID string `json:"x-obs-request-id"`
    OBSID2       string `json:"x-obs-id-2"`
}

//OBSInfo OBS信息
type OBSInfo struct {
    Version          string      `json:"Version"`
    ConfigurationID string      `json:"configurationId"`
    Bucket           BucketInfo `json:"bucket"`
    Object           ObjectInfo `json:"object"`
}

//BucketInfo 桶信息
type BucketInfo struct {
    Name          string       `json:"name"`
    OwnerIdentity UserIdentity `json:"ownerIdentity"`
    Bucket        string       `json:"bucket"`
}

//ObjectInfo 对象信息
type ObjectInfo struct {
    Key       string `json:"key"`
    Tag       string `json:"eTag"`
    Size      uint64 `json:"size"`
    VersionID string `json:"versionId"`
    Sequencer string `json:"sequencer"`
}

type Payload struct {
    ExecutionName string `json:"execution_name"`
    GraphName     string `json:"graph_name"`
    OBSMessages
    DynamicSource interface{}             `json:"dynamic_source"`
    Inputs        map[string]interface{} `json:"inputs"`
}
```