ALIBABA CLOUD

阿里云

实时计算（流计算）
OpenAPI

文档版本：20211227

阿里云

法律声明

阿里云提醒您在阅读或使用本文档之前仔细阅读、充分理解本法律声明各条款的内容。如果您阅读或使用本文档，您的阅读或使用行为将被视为对本声明全部内容的认可。

1. 您应当通过阿里云网站或阿里云提供的其他授权通道下载、获取本文档，且仅能用于自身的合法合规的业务活动。本文档的内容视为阿里云的保密信息，您应当严格遵守保密义务；未经阿里云事先书面同意，您不得向任何第三方披露本手册内容或提供给任何第三方使用。

2. 未经阿里云事先书面许可，任何单位、公司或个人不得擅自摘抄、翻译、复制本文档内容的部分或全部，不得以任何方式或途径进行传播和宣传。

3. 由于产品版本升级、调整或其他原因，本文档内容有可能变更。阿里云保留在没有任何通知或者提示下对本文档的内容进行修改的权利，并在阿里云授权通道中不时发布更新后的用户文档。您应当实时关注用户文档的版本变更并通过阿里云授权渠道下载、获取最新版的用户文档。

4. 本文档仅作为用户使用阿里云产品及服务的参考性指引，阿里云以产品及服务的“现状”、“有缺陷”和“当前功能”的状态提供本文档。阿里云在现有技术的基础上尽最大努力提供相应的介绍及操作指引，但阿里云在此明确声明对本文档内容的准确性、完整性、适用性、可靠性等不作任何明示或暗示的保证。任何单位、公司或个人因为下载、使用或信赖本文档而发生任何差错或经济损失的，阿里云不承担任何法律责任。在任何情况下，阿里云均不对任何间接性、后果性、惩戒性、偶然性、特殊性或刑罚性的损害，包括用户使用或信赖本文档而遭受的利润损失，承担责任（即使阿里云已被告知该等损失的可能性）。

5. 阿里云网站上所有内容，包括但不限于著作、产品、图片、档案、资讯、资料、网站架构、网站画面的安排、网页设计，均由阿里云和/或其关联公司依法拥有其知识产权，包括但不限于商标权、专利权、著作权、商业秘密等。非经阿里云和/或其关联公司书面同意，任何人不得擅自使用、修改、复制、公开传播、改变、散布、发行或公开发表阿里云网站、产品程序或内容。此外，未经阿里云事先书面同意，任何人不得为了任何营销、广告、促销或其他目的使用、公布或复制阿里云的名称（包括但不限于单独为或以组合形式包含“阿里云”、“Aliyun”、“万网”等阿里云和/或其关联公司品牌，上述品牌的附属标志及图案或任何类似公司名称、商号、商标、产品或服务名称、域名、图案标示、标志、标识或通过特定描述使第三方能够识别阿里云和/或其关联公司）。

6. 如若发现本文档存在任何错误，请与阿里云取得直接联系。

通用约定

格式	说明	样例
危险	该类警示信息将导致系统重大变更甚至故障，或者导致人身伤害等结果。	危险 重置操作将丢失用户配置数据。
警告	该类警示信息可能会导致系统重大变更甚至故障，或者导致人身伤害等结果。	警告 重启操作将导致业务中断，恢复业务时间约十分钟。
注意	用于警示信息、补充说明等，是用户必须了解的内容。	注意 权重设置为0，该服务器不会再接受新请求。
说明	用于补充说明、最佳实践、窍门等，不是用户必须了解的内容。	说明 您也可以通过按Ctrl+A选中全部文件。
>	多级菜单递进。	单击**设置**> **网络**> **设置网络类型**。
粗体	表示按键、菜单、页面名称等UI元素。	在**结果确认**页面，单击**确定**。
Courier字体	命令或代码。	执行 `cd /d C:/window` 命令，进入Windows系统文件夹。
斜体	表示参数、变量。	`bae log list --instanceid` *Instance_ID*
[] 或者 [a|b]	表示可选项，至多选择一个。	`ipconfig [-all|-t]`
{} 或者 {a|b}	表示必选项，至多选择一个。	`switch {active|stand}`

目录

1.简介

您可以使用实时计算的OpenAPI来操作您在实时计算产品上的所有项目作业，达到和使用WebUI操作相同的效果。

使用OpenAPI时，您需要先构造一个 `DefaultAcsClient` 类的客户端，传入相关配置，通过构造的客户端来发送和接受请求相关信息

所有的OpenAPI功能均成对出现，分为 `x-request` 和 `x-response` 两类， `request` 的OpenAPI负责接受参数，通过您创建的 `client` 客户端向实时计算发送请求，请求返回的结果是对应的 `response` 类，您可以通过get方法获取该类中存储的信息。

集群地域

集群地域和OpenAPI中 `regionID` 的对照关系，如下表所示。

地域	名称（regionID）
华东1	cn-hangzhou
华东2	cn-shanghai
华南1	cn-shenzhen
华北2	cn-beijing
华北3	cn-zhangjiakou
华东1金融云	cn-hangzhou-finance
华东2金融云	cn-shanghai-finance-1
华南1金融云	cn-shenzhen-finance-1
中国（香港）	cn-hongkong
日本（东京）	ap-northeast-1
亚太东南1（新加坡）	ap-southeast-1
亚太东南3（吉隆坡）	ap-southeast-3
美国东部1（弗吉尼亚）	us-east-1
欧洲中部1（法兰克福）	eu-central-1

机器型号

机器型号和OpenAPI中 `model` 的对应关系，如下表所示。

机器型号	名称（model）
4核16g	Ecs_4c16g

机器型号	名称（model）
8核32g	Ecs_8c32g
16核64g	Ecs_16c64g
24核96g	Ecs_24c96g
32核128g	Ecs_32c128g
56核224g	Ecs_56c224g
64核256g	Ecs_64c256g

2.OpenAPI概览

OpenAPI概览为您列举实时计算所支持的OpenAPI。

作业

OpenAPI	描述
CheckRawPlanJson	检测作业PlanJson的获取状态
CommitJob	提交作业
CreateJob	创建作业
DeleteJob	删除作业
GetJob	获取job信息
ListJob	搜索Job
OfflineJob	下线作业
StartJob	启动作业
UpdateJob	更新作业
ValidateJob	作业语法检查

文件夹

OpenAPI	描述
MVFolder	移动文件夹
CreateFolder	创建文件夹
DeleteFolder	删除文件夹
GetFolder	获取文件夹
ListChildFolder	获取子目录

实例

OpenAPI	描述
BatchGetInstanceRunSummary	批量获取实例运行信息
GetInstanceResource	获取运行实例使用资源
GetInstance	获取运行实例详情

OpenAPI	描述
GetInstanceCheckpoint	获取运行实例的Checkpoint
GetInstanceConfig	获取作业运行参数
GetInstanceDetail	获取实例运行的DAG图
GetInstanceExceptions	获取运行实例的Failover信息
GetInstanceFinalState	获取运行实例最终状态
GetInstanceMetric	获取运行实例的Metric信息
GetInstanceRunSummary	获取作业运行实例的运行概要
ListInstance	获取某个项目下所有的运行实例
ModifyInstanceState	修改实例状态
GetRawPlanJson	获取原始执行计划

Package

OpenAPI	描述
CreatePackage	获取Package信息
DeletePackage	删除Package
GetPackage	获得Package
ListPackage	搜索Package
GetRefPackageJob	获取引用特定Package的作业
UpdatePackage	更新Package

Queue

OpenAPI	描述
CreateQueue	创建Queue
DeleteQueue	删除Queue
BindQueue	绑定项目运行需要的Queue
UnbindQueue	解绑项目运行的Queue
GetClusterQueueInfo	获取集群中Queue的信息
ListProjectBindQueue	查询项目绑定的队列信息

OpenAPI	描述
ListProjectBindQueueResource	查询项目绑定的队列的资源信息

Cluster

OpenAPI	描述
CreateCluster	创建集群
ExpandCluster	后付费集群升级
DestroyCluster	注销集群
ShrinkCluster	集群缩容
ListCluster	查询已存在集群信息
ModifyMasterSpec	变更Master机型规格
GetClusterResource	获取集群资源状态信息
GetClusterEngineVersions	获取集群引擎版本
GetClusterDetails	获取集群详情

Project

OpenAPI	描述
CreateProject	创建项目
DeleteProject	删除项目
GetProject	获取项目详情
ListProject	查询您已有的项目

3.RAM资源授权

刚创建的子账号无法使用主账号的资源，您需要通过RAM授权的方式，赋予子账号操作主账号资源的权限。

默认情况下，您对自己创建的资源拥有完整的操作权限，可以使用本文档中列举的OpenAPI对资源进行操作。

> **说明** 如果您不需要跨账户授权和访问实例资源，您可以跳过该章节。

授权策略（Policy）

授权策略（Policy）描述授权的具体内容。授权内容主要包含效果（Effect）、资源（Resource）、对资源所授予的操作权限（Action）以及限制条件（Condition）这几个基本元素。以下内容以OpenAPI网关为例，为您介绍授权策略（Policy）。

- 系统授权策略

  OpenAPI网关已经预置了AliyunApiGatewayFullAccess和AliyunApiGatewayReadOnlyAccess两个系统权限，您可以在RAM控制台，**权限管理 > 权限策略管理**页面查看。

  - AliyunApiGatewayFullAccess：管理员权限，拥有主账号下OpenAPI分组、OpenAPI、流控策略、应用等所有资源的管理权限。
  - AliyunApiGatewayReadOnlyAccess：可以查看主账号下OpenAPI分组、OpenAPI、流控策略、应用等所有资源，但不可以操作。

- 自定义策略

  您可以在RAM控制台上**权限策略管理**页面，**自定义策略**查看已经定义好的自定义授权。

  您也可以根据需要自定义管理权限，支持更为精细化的授权。可以为某个操作或者某个资源授权。例如，GetUsers的编辑权限。自定义策略的查看、创建、修改和删除方法请参见授权策略管理。授权策略内容填写方法请参见权限策略基本元素、权限策略语法和结构。

- 示例

  允许所有的查看操作。

```
{
  "Version": "1",
  "Statement": [
    {
  "Action": "apigateway:Describe*",
      "Resource": "*",
      "Effect": "Allow"
    }
  ]
}
```

- Action（操作名称列表）格式

  "Action":"<service-name>:<action-name>"

  - service-name：阿里云产品名称，请填写 `apigateway` 。
  - action-name：OpenAPI接口名称，支持通配符 `*` 。

    ```
    "Action": "apigateway:Describe*" --表示所有的查询操作。
    "Action": "apigateway:*" --表示API网关所有操作。
    ```

- Resource（操作对象列表）
  - Resource通常指操作对象

    OpenAPI网关中的OpenAPI分组、流控策略、应用都被称为Resource，语法格式：acs:<service-name>:<region>:<account-id>:<relative-id>。其中：

    - acs：Alibaba Cloud Service的首字母缩写，表示阿里云的公共云平台。
    - service-name：阿里云产品名称，请填写 `apigateway` 。
    - region：地区信息，可以使用通配符号 `*` 来表示所有区域。
    - account-id：账号ID，例如， `1234567890123456` ，也可以使用 `*` 代替。
    - relative-id：与服务相关的资源描述部分，其语义由具体服务指定，该部分的格式支持树状结构（类似文件路径）。以OSS为例，表示一个OSS对象的格式为： `relative-id = “mybucket/dir1/object1.jpg”` 。
  - Resource语法

    acs:apigateway:$regionid:$accountid:apigroup/$groupId
  - Resource示例

    acs:apigateway:cn-hangzhou:195000****654:apigroup/i-xxxx

鉴权规则

以OpenAPI网关为例。

action-name	资源（Resource）
AbolishApi	acs:apigateway:$regionid:$accountid:apigroup/$groupId

4.调用方式

4.1. 签名机制

阿里云通过使用Access Key ID和Access Key Secret对称加密方法对请求发送者的身份进行验证。因此使用HTTP协议或HTTPS协议提交请求时，都需要在请求中包含签名（Signature）信息。

通过在阿里云官网申请，您可以获取由阿里云官网授予的 `Access Key ID` 和 `Access Key Secret` 。

> **注意** 您的密钥信息仅用于身份验证，请勿透传给无关人员使用。

- `Access Key ID` ：标识访问者的身份。
- `Access Key Secret` ：加密签名字符串和服务器端验证签名字符串的密钥。

> **说明** 阿里云提供了多种语言的SDK及第三方SDK，可以免去您对签名算法进行编码的工作。详情请参见阿里云开发工具包（SDK）。

签名操作

您在使用API访问阿里云时，需要按照以下步骤对请求进行签名处理。

1. 使用请求参数构造规范化的请求字符串（Canonicalized Query String）。
   i. 参数排序。按照参数名称的字典顺序对请求中所有的请求参数（包括除 `Signature` 以外的公共请求参数和接口的自定义参数）进行排序。

      > **说明** 当使用GET方法提交请求时的参数信息，即请求URI中的参数部分（即URI中 `?` 之后由 `&` 连接的部分）。

ii. 参数编码。 对排序之后的请求参数的名称和值分别用UTF-8字符集进行URL编码。编码的规则如下：

- 对于字符A~Z、a~z、0~9以及中划线（-）、下划线（_）、英文句号（.）、波浪号（~）不编码。
- 对于其它字符编码成 `%XY` 的格式，其中 `XY` 是字符对应ASCII码的16进制表示。例如，英文的双引号（”）对应的编码为 `%22` 。
- 对于扩展的UTF-8字符，编码成 `%XY%ZA…` 的格式。
- 英文空格要编码为 `%20` ，而不是加号（+）。

该编码方式和通常采用的 `application/x-www-form-urlencoded` MIME格式编码算法的实现（例如Java标准库中的 `java.net.URLEncoder` ）相似，但又有所不同。实现时，可以先用标准库的方式进行编码，然后把编码后的字符串中加号（+）替换成 `%20` 、星号（*）替换成 `%2A` 、 `%7E` 替换回波浪号（~），即可得到上述规则描述的编码字符串。该算法可以用下面的percentEncode方法实现。

```
private static final String ENCODING = "UTF-8";
private static String percentEncode(String value) throws UnsupportedEncodingExcepti
on {
return value != null ? URLEncoder.encode(value, ENCODING).replace("+", "%20").repla
ce("*", "%2A").replace("%7E", "~") : null;
}
```

iii. 将编码后的参数名称和值用英文等号（=）进行连接。

iv. 将等号连接得到的参数组合按照参数排序依次使用（&）符号连接，即得到规范化请求字符串。

2. 将上一步构造的规范化字符串按照下面的规则构造成待签名的字符串。

```
StringToSign= HTTPMethod + “&” + percentEncode(“/”) + ”&” +
        percentEncode(CanonicalizedQueryString)
```

其中：

- `HTTPMethod` 是提交请求用的HTTP方法，例如GET。
- `percentEncode(“/”)` 是按照参数排序中描述的URL编码规则对字符 `/` 进行编码得到的值，即 `%2F` 。
- `percentEncode(CanonicalizedQueryString)` 是对步骤1中构造的规范化请求字符串按参数编码中描述的URL编码规则编码后得到的字符串。

3. 按照RFC 2104的定义，计算待签名字符串StringToSign的HMAC值。

> **说明** 计算签名时使用的Key就是您持有的Access Key Secret加上一个 `&` 字符（ASCII:38），使用的哈希算法是SHA1。

4. 按照Base64编码规则把上面的HMAC值编码成字符串，即得到签名值（Signature）。
5. 将得到的签名值作为Signature参数添加到请求参数中，即完成对请求签名的过程。

> **说明** 得到的签名值在作为最后的请求参数值提交给ECS服务器时，要和其它参数一样，按照RFC3986的规则进行URL编码。

示例

以DescribeRegions为例，假设使用的 `Access Key Id` 为 `testid` ， `Access Key Secret` 为 `testsecret` ，则签名前的请求URL如下所示。

```
http://ecs.aliyuncs.com/
?TimeStamp=2016-02-23T12:46:24Z
&Format=XML
&AccessKeyId=testid
&Action=DescribeRegions
&SignatureMethod=HMAC-SHA1
&SignatureNonce=3ee8c1b8-83d3-44af-a94f-4e0ad82fd6cf
&Version=2014-05-26
&SignatureVersion=1.0
```

计算得到的待签名字符串 `StringToSign` 如下所示。

```
GET
&%2F
&AccessKeyId%3Dtestid
&Action%3DDescribeRegions
&Format%3DXML&SignatureMethod%3DHMAC-SHA1
&SignatureNonce%3D3ee8c1b8-83d3-44af-a94f-4e0ad82fd6cf
&SignatureVersion%3D1.0
&TimeStamp%3D2016-02-23T12%253A46%253A24Z
&Version%3D2014-05-26
```

因为 `Access Key Secret` 为 `testsecret` ，所以用于计算HMAC的Key为 `testsecret&` ，计算得到的签名值为 `CT9X0VtwR86fNWSnsc6v8YGOjuE=` 。

将签名作为Signature参数加入到URL请求中，最后得到的URL如下所示。

```
http://ecs.aliyuncs.com/
?SignatureVersion=1.0
&Action=DescribeRegions
&Format=XML
&SignatureNonce=3ee8c1b8-83d3-44af-a94f-4e0ad82fd6cf
&Version=2014-05-26
&AccessKeyId=testid
&Signature=CT9X0VtwR86fNWSnsc6v8YGOjuE%3D
&SignatureMethod=HMAC-SHA1
&TimeStamp=2016-02-23T12%3A46%3A24Z
```

4.2. 请求结构

本文为您介绍OpenAPI服务的请求结构。

服务地址

API 的服务接入地址，如下所示：

地域	服务地址
杭州	`foas.aliyuncs.com`
cn-shanghai：上海	`foas.[RegionId].aliyuncs.com`

通信协议

为了获得更高的安全性，仅支持使用HTTPS通道发送OpenAPI请求。

字符编码

请求及返回结果都使用UTF-8 字符集进行编码。

4.3. 返回结果

本文为您介绍OpenAPI服务的返回结果。

返回结果格式

调用OpenAPI服务后返回数据采用统一格式：

- 返回的HTTP状态码为 `2xx` ，代表调用成功。
- 返回的HTTP状态码为 `4xx` 或 `5xx` ，代表调用失败。

> **说明** 本文档中的返回示例为了便于用户查看，做了格式化处理，实际返回结果是没有进行换行、缩进等处理的。

返回结果成功

调用成功返回的数据格式主要有XML和JSON两种，外部系统可以在请求时传入参数来制定返回的数据格式，默认为XML格式。返回结果成功示例如下：

```
{}
```

返回结果错误

调用接口出错后，将不会返回结果数据。调用方可根据每个接口对应的错误码以及下述 公共错误码来定位错误原因。 当调用出错时， `HTTP` 请求返回一个 `4xx` 或 `5xx` 的 `HTTP` 状态码。返回的消息体中是具体的错误代码及错误信息。另外还包含一个全局唯一的请求ID： `RequestId` 和一个您该次请求访问的站点ID： `HostId` 。在调用方找不到错误原因时，可以联系阿里云客服，并提供该 `HostId` 和 `RequestId` ，以便尽快为您解决问题。返回结果错误示例如下：

```
null
```

公共错误码

请参见：公共错误码表。

4.4. 公共参数

公共参数指的是所有接口调用都需要用到的参数，包含公共请求参数和公共返回参数两种。

公共请求参数

公共请求参数是指每个接口都需要使用到的请求参数。

名称	类型	是否必选	描述
Format	String	否	返回值的类型，支持JSON与XML。默认为XML。
Version	String	是	OpenAPI版本号，为日期形式：`YYYY-MM-DD`，本版本对应为`2018-11-11`。
AccessKeyId	String	是	阿里云颁发给用户的访问服务所用的密钥ID。
Signature	String	是	签名结果串，关于签名的计算方法，请参见签名机制。
SignatureMethod	String	是	签名方式，目前支持`HMAC-SHA1`。
Timestamp	String	是	请求的时间戳。日期格式按照 ISO8601标准表示，并需要使用UTC时间。格式为：`YYYY-MM-DDThh:mm:ssZ`，例如，`2014-05-26T12:00:00Z`（为北京时间 2014年5月26日20点0分0秒）。
SignatureVersion	String	是	签名算法版本，目前版本是`1.0`。
SignatureNonce	String	是	唯一随机数，用于防止网络重放攻击。您在不同请求间要使用不同的随机数值
ResourceOwnerAccount	String	否	本次OpenAPI请求访问到的资源拥有者账户，即登录用户名。 此参数的使用方法，详见借助RAM实现子账号对主账号的资源访问 （只能在RAM中可对ECS资源进行授权的Action中才能使用此参数，否则访问会被拒绝）。

公共请求参数示例

```
https://foas.aliyuncs.com/
?Format=xml
&Version=2018-11-11
&Signature=Pc5WB8gokVn0xfeu%2FZV%2BiNM1dgI%3D
&SignatureMethod=HMAC-SHA1
&SignatureNonce=15215528852396
&SignatureVersion=1.0
&AccessKeyId=key-test
&Timestamp=2012-06-01T12:00:00Z
…
```

公共返回参数

您发送的每次接口调用请求，无论成功与否，系统都会返回一个唯一识别码 `RequestId` 。

- XML示例

```
<?xml version="1.0" encoding="UTF-8"?>
<!--结果的根结点-->
<接口名称+Response>
<!--返回请求标签-->
<RequestId>4C467B38-3910-447D-87BC-AC049166F216</RequestId>
<!--返回结果数据-->
</接口名称+Response>
```

- JSON示例

```
{
"RequestId": "4C467B38-3910-447D-87BC-AC049166F216",
/* 返回结果数据 */
}
```

5.实例

5.1. BatchGetInstanceRunSummary

您可以通过BatchGetInstanceRunSummary接口获取指定项目下，指定类别的作业的Job的运行情况。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
GET /api/v2/projects/[projectName]/runsummary HTTP/1.1
```

请求参数

名称	类型	位置	是否必选	示例值	描述
jobNames	String	Query	是	job1,job2	作业名称，多个名称时使用逗号（,）分隔。
jobType	String	Query	是	FLINK_STREAM	作业类型： • FLINK_BATCH：批作业使用。 • FLINK_STREAM：流作业使用。
projectName	String	Path	是	project1	项目名称。
RegionId	String	Header	否	cn-hangzhou	区域ID。

返回数据

名称	类型	示例值	描述
RequestId	String	1CA9A8E9-F398-493F-83CA-A92A38501B47	请求ID。
RunSummarys	Array of RunSummary		作业运行状态详情。
RunSummary			

名称	类型	示例值	描述
ActualState	String	RUNNING	返回作业的实际运行状态： • WAITING：等待。 • RUNNING：正在运行。 • PAUSED：暂停。 • TERMINATED：停止。 • SUCCESS：成功（批作业）。 • FAILED：失败（批作业）。
EngineJobHandler	String	application_1561366511517_113805|a10928cec91de395ff3d7593f7ff46c5	YARN中为作业分配的名称：ApplicationID|JobID（JM分配的ID）。
ExpectState	String	RUNNING	作业期望状态： • WAITING：等待。 • RUNNING：正在运行。 • PAUSED：暂停。 • TERMINATED：停止。 • SUCCESS：成功（批作业）。 • FAILED：失败（批作业）。
Id	Long	214650	运行实例ID。
InputDelay	Long	0	业务延时。
JobName	String	ss2	作业名称，多个逗号（,）分隔。
LastErrorMessage	String	error	上次错误信息。
LastErrorTime	Long	1548397575000	上次出现错误时间（时间戳，精确到毫秒）。

示例

请求示例

```
/api/v2/projects/project1/runsummary
```

正常返回示例

`XML` 格式

```
<RequestId>1CA9A8E9-F398-493F-83CA-A92A38501B47</RequestId>
<RunSummarys>
    <RunSummary>
        <ActualState>RUNNING</ActualState>
        <EngineJobHandler>application_1561366511517_113805|a10928cec91de395ff3d7593f7ff46c5
</EngineJobHandler>
        <ExpectState>RUNNING</ExpectState>
        <JobName>ss2</JobName>
        <Id>214650</Id>
        <InputDelay>0</InputDelay>
    </RunSummary>
</RunSummarys>
```

`JSON` 格式

```
{
 "RequestId": "1CA9A8E9-F398-493F-83CA-A92A38501B47",
 "RunSummarys": {
  "RunSummary": [
   {
    "ActualState": "RUNNING",
    "EngineJobHandler": "application_1561366511517_113805|a10928cec91de395ff3d7593f7ff46c5"
,
    "ExpectState": "RUNNING",
    "JobName": "ss2",
    "Id": 214650,
    "InputDelay": 0
   }
  ]
 }
}
```

5.2. GetInstance

您可以通过GetInstance获取运行作业的全部信息，未运行的作业调用该接口会失败。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
GET /api/v2/projects/[projectName]/jobs/[jobName]/instances/[instanceId] HTTPS
```

请求参数

名称	类型	是否必选	示例值	描述
instanceId	Long	是	123	实例ID，可以通过ListInstance接口或者StartJob接口获取。
jobName	String	是	job1	作业名称
projectName	String	是	project1	项目名称
RegionId	String	是	cn-hangzhou-pre	区域ID **说明** 公共云用户请忽略此参数。

返回数据

名称	类型	示例值	描述
Instance	Struct		Instance详情
ActualState	String	RUNNING	Instance的实际状态： • WAITING：等待。 • RUNNING：运行中。 • PAUSED：暂停。 • TERMINATED：停止。 • SUCCESS：成功（批作业）。 • FAILED：失败（批作业）。
ApiType	String	DATASTREAM	作业类型： • DATASTREAM：Datastream作业。 • SQL：SQL作业。
AutoScaleParams	String	"{\"isOnOff\":false,\"strategyConfigure\":null,\"maxCU\":null}	作业上线时设置的AutoScale参数。
ClusterId	String	rcmp9x37ztfb63g1x7lt****	集群ID

名称	类型	示例值	描述
Code	String	--完整主类名，必填，例如com.alibaba.realtimecompute.DatastreamExample\nblink.main.class=Hbase_Demo.HbaseDemo\n\n--包含完整主类名的JAR包资源名称，多个JAR包时必填，例如blink_datastream.jar\n--blink.main.jar=hbase_demo-1.0-snapshot.jar \n\n--默认state backend配置，当作业代码没有显式声明时生效\nstate.backend.type=niagara\nstate.backend.niagara.ttl.ms=129600000\n\n--默认Checkpoint配置，当作业代码没有显式声明时生效\nblink.checkpoint.interval.ms=180000\n\n--默认启用项目参数\n--enable.project.config=true	Instance的运行代码： • SQL作业返回SQL代码。 • Datastream作业返回Datastream配置。
EndTime	Long	1548397575000	作业结束时间，未结束作业返回空值。
EngineJobHandler	String	application_1597654819348_****|405a6d40d7f4f2618ed532359887****	Instance在YARN中的名称，格式为：ApplicaitonID|jobID（JobManager分配的jobid）。
EngineVersion	String	blink-3.4.4	Blink版本
ExpectState	String	RUNNING	期望状态： • RUNNING：运行中。 • PAUSED：暂停。 • TERMINATED：停止。 • SUCCESS：成功（批作业）。 • FAILED：失败（流作业）。

名称	类型	示例值	描述
Id	Long	412536	InstanceID
InputDelay	Long	-99999999	业务延迟，单位为毫秒。
JobName	String	job1	作业名称
JobType	String	FLINK_STREAM	作业类型： • FLINK_STREAM：流作业。 • FLINK_BATCH：批作业。
LastErrorMessage	String	error	最新的报错信息
LastErrorTime	Long	1548397575000	最新的报错时间
LastOperateTime	Long	1599794334000	最新的操作时间
LastOperator	String	239960377092765296	最新的操作人
Packages	String	hbase_demo-1.0-snapshot-shaded.jar	实例引用的包名称，多个Package使用逗号（,）分隔，未引用为空。
PlanJson	String	{a:b}	执行计划
ProjectName	String	project1	项目名称

名称	类型	示例值	描述
Properties	String	#checkpoint模式：EXACTLY_ONCE 或者 AT_LEAST_ONCE\n#blink.checkpoint.mode=EXACTLY_ONCE\n\n#checkpoint间隔时间，单位毫秒\n#blink.checkpoint.interval.ms=180000\n\n#rocksdb的数据生命周期，单位毫秒\n#state.backend.rocksdb.ttl.ms=129600000\n\n#启用项目参数\n#enable.project.config=true\nblink.job.timeZone=Asia/Shanghai	作业运行参数
QueueName	String	root.project1	队列名称
StartTime	Long	1599794335000	开始时间
Starter	String	239960377092765296	启动人
RequestId	String	B2F5A5C5-5CCC-45C8-B884-7152CC345EFA	请求ID

示例

请求示例

```
/api/v2/projects/project1/jobs/job1/instances/123
```

正常返回示例

`XML` 格式

```
<RequestId>9B7BD33D-5F3A-4C4C-BCF5-AD686CAD2370</RequestId>
<Instance>
    <ExpectState>RUNNING</ExpectState>
    <EngineVersion>blink-3.4.4</EngineVersion>
    <ProjectName>project1</ProjectName>
    <ClusterId>rcmp9x37ztfb63g1x7lt****</ClusterId>
    <PlanJson/>
    <JobName>job1</JobName>
    <StartTime>1599794335000</StartTime>
    <Properties>#checkpoint模式：EXACTLY_ONCE 或者 AT_LEAST_ONCE
#blink.checkpoint.mode=EXACTLY_ONCE
#checkpoint间隔时间，单位毫秒
#blink.checkpoint.interval.ms=180000
#rocksdb的数据生命周期，单位毫秒
#state.backend.rocksdb.ttl.ms=129600000
#启用项目参数
#enable.project.config=true
blink.job.timeZone=Asia/Shanghai</Properties>
    <Starter>239960377092765296</Starter>
    <InputDelay>-99999999</InputDelay>
    <Code>--完整主类名，必填，例如com.alibaba.realtimecompute.DatastreamExample
blink.main.class=Hbase_Demo.HbaseDemo
--包含完整主类名的JAR包资源名称，多个JAR包时必填，例如blink_datastream.jar
--blink.main.jar=hbase_demo-1.0-snapshot.jar
--默认state backend配置，当作业代码没有显式声明时生效
state.backend.type=niagara
state.backend.niagara.ttl.ms=129600000
--默认Checkpoint配置，当作业代码没有显式声明时生效
blink.checkpoint.interval.ms=180000
--默认启用项目参数
--enable.project.config=true</Code>
    <ActualState>RUNNING</ActualState>
    <EngineJobHandler>application_1597654819348_0011|405a6d40d7f4f2618ed532359887d344</Engi
neJobHandler>
    <LastOperator>239960377092765296</LastOperator>
    <JobType>FLINK_STREAM</JobType>
    <Packages>hbase_demo-1.0-snapshot-shaded.jar</Packages>
    <AutoScaleParams>{"isOnOff":false,"strategyConfigure":null,"maxCU":null}</AutoScalePara
ms>
    <ApiType>DATASTREAM</ApiType>
    <Id>412536</Id>
    <LastOperateTime>1599794334000</LastOperateTime>
    <QueueName>root.project1</QueueName>
</Instance>
```

`JSON` 格式

```
{
 "RequestId": "9B7BD33D-5F3A-4C4C-BCF5-AD686CAD2370",
 "Instance": {
  "ExpectState": "RUNNING",
  "EngineVersion": "blink-3.4.4",
  "ProjectName": "project1",
  "ClusterId": "rcmp9x37ztfb63g1x7lt****",
  "PlanJson": "",
  "JobName": "job1",
  "StartTime": 1599794335000,
  "Properties": "#checkpoint模式：EXACTLY_ONCE 或者 AT_LEAST_ONCE\n#blink.checkpoint.mode=EXACTLY_ONCE\n\n#checkpoint间隔时间，单位毫秒\n#blink.checkpoint.interval.ms=180000\n\n#rocksdb的数据生命周期，单位毫秒\n#state.backend.rocksdb.ttl.ms=129600000\n\n#启用项目参数\n#enable.project.config=true\nblink.job.timeZone=Asia/Shanghai",
  "Starter": "239960377092765296",
  "InputDelay": -99999999,
  "Code": "--完整主类名，必填，例如com.alibaba.realtimecompute.DatastreamExample\nblink.main.class=Hbase_Demo.HbaseDemo\n\n--包含完整主类名的JAR包资源名称，多个JAR包时必填，例如blink_datastream.jar\n--blink.main.jar=hbase_demo-1.0-snapshot.jar \n\n--默认state backend配置，当作业代码没有显式声明时生效\nstate.backend.type=niagara\nstate.backend.niagara.ttl.ms=129600000\n\n--默认Checkpoint配置，当作业代码没有显式声明时生效\nblink.checkpoint.interval.ms=180000\n\n--默认启用项目参数\n--enable.project.config=true",
  "ActualState": "RUNNING",
  "EngineJobHandler": "application_1597654819348_0011|405a6d40d7f4f2618ed532359887d344",
  "LastOperator": "239960377092765296",
  "JobType": "FLINK_STREAM",
  "Packages": "hbase_demo-1.0-snapshot-shaded.jar",
  "AutoScaleParams": "{\"isOnOff\":false,\"strategyConfigure\":null,\"maxCU\":null}",
  "ApiType": "DATASTREAM",
  "Id": 412536,
  "LastOperateTime": 1599794334000,
  "QueueName": "root.project1"
 }
}
```


5.3. GetInstanceResource

您可以通过GetInstanceResource OpenAPI获取运行实例信息和已使用的CPU和Memory等信息。未运行的作业调用GetInstanceResource OpenAPI会报错。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
GET /api/v2/projects/[projectName]/jobs/[jobName]/instances/[instanceId]/resource HTTP/1.1
```

请求参数

名称	类型	位置	是否必选	示例值	描述
instanceId	Long	Path	是	403936	实例ID： • 流作业：填写实际运行的实例ID或直接填写-1（当前运行的实例）。 • 批作业：可以通过ListInstance和StartJob等接口获得。
jobName	String	Path	是	test_0812	作业名称。
projectName	String	Path	是	bayes_team	项目名称。
RegionId	String	Header	否	cn-hangzhou	区域ID。

返回数据

名称	类型	示例值	描述
RequestId	String	1A90ED7D-6A31-4385-8372-B73A617A0D50	请求ID。
Resource	Struct		资源详情。
AllocatedMB	Long	2816	作业申请的内存。
AllocatedVirtualCores	Long	125	作业申请的Vcore。
TotalMB	Long	1024	实际运行使用的内存。
TotalVirtualCores	Long	50	实际运行使用的Vcore。

示例

请求示例

```
/api/v2/projects/bayes_team/jobs/test_0812/instances/412532/resource
```

正常返回示例

`XML` 格式

```
<RequestId>1A90ED7D-6A31-4385-8372-B73A617A0D50</RequestId>
<Resource>
    <AllocatedMB>2816</AllocatedMB>
    <AllocatedVirtualCores>125</AllocatedVirtualCores>
</Resource>
```

`JSON` 格式

```
{
 "RequestId": "1A90ED7D-6A31-4385-8372-B73A617A0D50",
 "Resource": {
  "AllocatedMB": 2816,
  "AllocatedVirtualCores": 125
 }
}
```

5.4. GetInstanceCheckpoint

您可以通过GetInstanceCheckpoint API获取正在运行实例作业的Checkpoint，非运行状态的作业调用此API会返回错误。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
GET /api/v2/projects/[projectName]/jobs/[jobName]/instances/[instanceId]/checkpoints HTTP/1
.1
```

请求参数

名称	类型	位置	是否必选	示例值	描述
instanceId	Long	Path	是	403936	实例ID： • 流作业：填写实际运行的实例ID或直接填写-1（当前运行的实例）。 • 批作业：可以通过ListInstance、StartJob等接口获得。
jobName	String	Path	是	test_0812	作业名称。

名称	类型	位置	是否必选	示例值	描述
projectName	String	Path	是	bayes_team	项目名称。
RegionId	String	Header	是	cn-hangzhou	区域ID。

返回数据

名称	类型	示例值	描述
		{\"counts\":{\"restored\":1,\"total\":61440,\"in_progress\":0,\"completed\":61440,\"failed\":0},\"summary\":{\"state_size\":{\"min\":4384,\"max\":4384,\"avg\":4384},\"end_to_end_duration\":{\"min\":9,\"max\":981,\"avg\":15},\"alignment_buffered\":{\"min\":0,\"max\":0,\"avg\":0}},\"latest\":{\"completed\":{\"@class\":\"completed\",\"id\":61440,\"status\":\"COMPLETED\",\"is_savepoint\":false,\"trigger_timestamp\":1610505450869,\"latest_ack_timestamp\":1610505450883,\"state_size\":4384,\"full_state_size\":4384,\"end_to_end_duration\":14,\"alignment_buffered\":0,\"num_subtasks\":1,\"num_acknowledged_subtasks\":1,\"tasks\":{},\"external_path\":\"hdfs://et2cloudprod/blink_rest_server/root.bayes_team/test_0812/chk-61440\",\"discarded\":false},\"savepoint\":null,\"failed\":n	

名称	类型	示例值	描述
		ull,\"restored\":{\"id\":7965,\"restore_timestamp\":1600879950869,\"is_savepoint\":false,\"external_path\":\"hdfs://et2cloudprod/blink_rest_server/root.bayes_team/test_0812/chk-7965\"}},\"history\":[{\"@class\":\"completed\",\"id\":61440,\"status\":\"COMPLETED\",\"is_savepoint\":false,\"trigger_timestamp\":1610505450869,\"latest_ack_timestamp\":1610505450883,\"state_size\":4384,\"full_state_size\":4384,\"end_to_end_duration\":14,\"alignment_buffered\":0,\"num_subtasks\":1,\"num_acknowledged_subtasks\":1,\"tasks\":{},\"external_path\":\"hdfs://et2cloudprod/blink_rest_server/root.bayes_team/test_0812/chk-61440\",\"discarded\":false},{\"@class\":\"completed\",\"id\":61439,\"status\":\"COMPLETED\",\"is_savepoint\":false,\"trigger_timestamp\":1610505270869,\"latest_ack_timestamp\":1610505270888,\"state_size\":4384,\"full_state_size\":4384,\"end_to_end_duration\":19,\"alignment_buffered\":0,\"num_subtasks\":1,\"num_acknowledged_subtasks\":1,\"tasks\":{},\"external_path\":\"hdfs://et2cloudprod/blink_rest_server/root.bayes_team/t	

名称	类型	示例值	描述
Checkpoints	String	est_0812/chk-61439\",\"discarded\":true},{\"@class\":\"completed\",\"id\":61438,\"status\":\"COMPLETED\",\"is_savepoint\":false,\"trigger_timestamp\":1610505090869,\"latest_ack_timestamp\":1610505090882,\"state_size\":4384,\"full_state_size\":4384,\"end_to_end_duration\":13,\"alignment_buffered\":0,\"num_subtasks\":1,\"num_acknowledged_subtasks\":1,\"tasks\":{},\"external_path\":\"hdfs://et2cloudprod/blink_rest_server/root.bayes_team/test_0812/chk-61438\",\"discarded\":true},{\"@class\":\"completed\",\"id\":61437,\"status\":\"COMPLETED\",\"is_savepoint\":false,\"trigger_timestamp\":1610504910869,\"latest_ack_timestamp\":1610504910882,\"state_size\":4384,\"full_state_size\":4384,\"end_to_end_duration\":13,\"alignment_buffered\":0,\"num_subtasks\":1,\"num_acknowledged_subtasks\":1,\"tasks\":{},\"external_path\":\"hdfs://et2cloudprod/blink_rest_server/root.bayes_team/test_0812/chk-61437\",\"discarded\":true},{\"@class\":\"completed\",\"id\":61436,\"status\":\"COMPLETED\",\"is_savepoint\":false,\"trigger_	Checkpoint详情。

名称	类型	示例值	描述
		timestamp\":1610504730869,\"latest_ack_timestamp\":1610504730882,\"state_size\":4384,\"full_state_size\":4384,\"end_to_end_duration\":13,\"alignment_buffered\":0,\"num_subtasks\":1,\"num_acknowledged_subtasks\":1,\"tasks\":{},\"external_path\":\"hdfs://et2cloudprod/blink_rest_server/root.bayes_team/test_0812/chk-61436\",\"discarded\":true},{\"@class\":\"completed\",\"id\":61435,\"status\":\"COMPLETED\",\"is_savepoint\":false,\"trigger_timestamp\":1610504550869,\"latest_ack_timestamp\":1610504550883,\"state_size\":4384,\"full_state_size\":4384,\"end_to_end_duration\":14,\"alignment_buffered\":0,\"num_subtasks\":1,\"num_acknowledged_subtasks\":1,\"tasks\":{},\"external_path\":\"hdfs://et2cloudprod/blink_rest_server/root.bayes_team/test_0812/chk-61435\",\"discarded\":true},{\"@class\":\"completed\",\"id\":61434,\"status\":\"COMPLETED\",\"is_savepoint\":false,\"trigger_timestamp\":1610504370869,\"latest_ack_timestamp\":1610504370885,\"state_size\":4384,\"full_state_size\":4384,\"end_to_end_duration\":16,\"alignment_b	

名称	类型	示例值	描述
		uffered\":0,\"num_subtasks\":1,\"num_acknowledged_subtasks\":1,\"tasks\":{},\"external_path\":\"hdfs://et2cloudprod/blink_rest_server/root.bayes_team/test_0812/chk-61434\",\"discarded\":true},{\"@class\":\"completed\",\"id\":61433,\"status\":\"COMPLETED\",\"is_savepoint\":false,\"trigger_timestamp\":1610504190869,\"latest_ack_timestamp\":1610504190882,\"state_size\":4384,\"full_state_size\":4384,\"end_to_end_duration\":13,\"alignment_buffered\":0,\"num_subtasks\":1,\"num_acknowledged_subtasks\":1,\"tasks\":{},\"external_path\":\"hdfs://et2cloudprod/blink_rest_server/root.bayes_team/test_0812/chk-61433\",\"discarded\":true},{\"@class\":\"completed\",\"id\":61432,\"status\":\"COMPLETED\",\"is_savepoint\":false,\"trigger_timestamp\":1610504010869,\"latest_ack_timestamp\":1610504010896,\"state_size\":4384,\"full_state_size\":4384,\"end_to_end_duration\":27,\"alignment_buffered\":0,\"num_subtasks\":1,\"num_acknowledged_subtasks\":1,\"tasks\":{},\"external_path\":\"hdfs://et2cloudprod/blink_rest_server/root.bayes_team/t	

名称	类型	示例值	描述
		est_0812/chk-61432\",\"discarded\":true},{\"@class\":\"completed\",\"id\":61431,\"status\":\"COMPLETED\",\"is_savepoint\":false,\"trigger_timestamp\":1610503830869,\"latest_ack_timestamp\":1610503830890,\"state_size\":4384,\"full_state_size\":4384,\"end_to_end_duration\":21,\"alignment_buffered\":0,\"num_subtasks\":1,\"num_acknowledged_subtasks\":1,\"tasks\":{},\"external_path\":\"hdfs://et2cloudprod/blink_rest_server/root.bayes_team/test_0812/chk-	
RequestId	String	A0623FAD-D4D7-4DFA-B126-D730C84338ED	请求ID。

示例

请求示例

```
/api/v2/projects/bayes_team/jobs/test_0812/instances/403936/checkpoints
```

正常返回示例

`XML` 格式

```
<RequestId>A0623FAD-D4D7-4DFA-B126-D730C84338ED</RequestId>
<Checkpoints>{"counts":{"restored":1,"total":61440,"in_progress":0,"completed":61440,"failed":0},"summary":{"state_size":{"min":4384,"max":4384,"avg":4384},"end_to_end_duration":{"min":9,"max":981,"avg":15},"alignment_buffered":{"min":0,"max":0,"avg":0}},"latest":{"completed":{"@class":"completed","id":61440,"status":"COMPLETED","is_savepoint":false,"trigger_timestamp":1610505450869,"latest_ack_timestamp":1610505450883,"state_size":4384,"full_state_size":4384,"end_to_end_duration":14,"alignment_buffered":0,"num_subtasks":1,"num_acknowledged_subtasks":1,"tasks":{},"external_path":"hdfs://et2cloudprod/blink_rest_server/root.bayes_team/test_0812/chk-61440","discarded":false},"savepoint":null,"failed":null,"restored":{"id":7965,"restore_timestamp":1600879950869,"is_savepoint":false,"external_path":"hdfs://et2cloudprod/blink_rest_server/root.bayes_team/test_0812/chk-7965"}},"history":[{"@class":"completed","id":61440,"status":"COMPLETED","is_savepoint":false,"trigger_timestamp":1610505450869,"latest_ack_timestamp":1610505450883,"state_size":4384,"full_state_size":4384,"end_to_end_duration":14,"alignment_buffered":0,"num_subtasks":1,"num_acknowledged_subtasks":1,"tasks":{},"external_path":"hdfs://et2cloudprod/blink_rest_server/root.bayes_team/test_0812/chk-61440","discarded":false},{"@class":"completed","id":61439,"status":"COMPLETED","is_savepoint":false,"trigger_timestamp":1610505270869,"latest_ack_timestamp":1610505270888,"state_size":
```

4384,"full_state_size":4384,"end_to_end_duration":19,"alignment_buffered":0,"num_subtasks":1,"num_acknowledged_subtasks":1,"tasks":{},"external_path":"hdfs://et2cloudprod/blink_rest_server/root.bayes_team/test_0812/chk-61439","discarded":true},{"@class":"completed","id":61438,"status":"COMPLETED","is_savepoint":false,"trigger_timestamp":1610505090869,"latest_ack_timestamp":1610505090882,"state_size":4384,"full_state_size":4384,"end_to_end_duration":13,"alignment_buffered":0,"num_subtasks":1,"num_acknowledged_subtasks":1,"tasks":{},"external_path":"hdfs://et2cloudprod/blink_rest_server/root.bayes_team/test_0812/chk-61438","discarded":true},{"@class":"completed","id":61437,"status":"COMPLETED","is_savepoint":false,"trigger_timestamp":1610504910869,"latest_ack_timestamp":1610504910882,"state_size":4384,"full_state_size":4384,"end_to_end_duration":13,"alignment_buffered":0,"num_subtasks":1,"num_acknowledged_subtasks":1,"tasks":{},"external_path":"hdfs://et2cloudprod/blink_rest_server/root.bayes_team/test_0812/chk-61437","discarded":true},{"@class":"completed","id":61436,"status":"COMPLETED","is_savepoint":false,"trigger_timestamp":1610504730869,"latest_ack_timestamp":1610504730882,"state_size":4384,"full_state_size":4384,"end_to_end_duration":13,"alignment_buffered":0,"num_subtasks":1,"num_acknowledged_subtasks":1,"tasks":{},"external_path":"hdfs://et2cloudprod/blink_rest_server/root.bayes_team/test_0812/chk-61436","discarded":true},{"@class":"completed","id":61435,"status":"COMPLETED","is_savepoint":false,"trigger_timestamp":1610504550869,"latest_ack_timestamp":1610504550883,"state_size":4384,"full_state_size":4384,"end_to_end_duration":14,"alignment_buffered":0,"num_subtasks":1,"num_acknowledged_subtasks":1,"tasks":{},"external_path":"hdfs://et2cloudprod/blink_rest_server/root.bayes_team/test_0812/chk-61435","discarded":true},{"@class":"completed","id":61434,"status":"COMPLETED","is_savepoint":false,"trigger_timestamp":1610504370869,"latest_ack_timestamp":1610504370885,"state_size":4384,"full_state_size":4384,"end_to_end_duration":16,"alignment_buffered":0,"num_subtasks":1,"num_acknowledged_subtasks":1,"tasks":{},"external_path":"hdfs://et2cloudprod/blink_rest_server/root.bayes_team/test_0812/chk-61434","discarded":true},{"@class":"completed","id":61433,"status":"COMPLETED","is_savepoint":false,"trigger_timestamp":1610504190869,"latest_ack_timestamp":1610504190882,"state_size":4384,"full_state_size":4384,"end_to_end_duration":13,"alignment_buffered":0,"num_subtasks":1,"num_acknowledged_subtasks":1,"tasks":{},"external_path":"hdfs://et2cloudprod/blink_rest_server/root.bayes_team/test_0812/chk-61433","discarded":true},{"@class":"completed","id":61432,"status":"COMPLETED","is_savepoint":false,"trigger_timestamp":1610504010869,"latest_ack_timestamp":1610504010896,"state_size":4384,"full_state_size":4384,"end_to_end_duration":27,"alignment_buffered":0,"num_subtasks":1,"num_acknowledged_subtasks":1,"tasks":{},"external_path":"hdfs://et2cloudprod/blink_rest_server/root.bayes_team/test_0812/chk-61432","discarded":true},{"@class":"completed","id":61431,"status":"COMPLETED","is_savepoint":false,"trigger_timestamp":1610503830869,"latest_ack_timestamp":1610503830890,"state_size":4384,"full_state_size":4384,"end_to_end_duration":21,"alignment_buffered":0,"num_subtasks":1,"num_acknowledged_subtasks":1,"tasks":{},"external_path":"hdfs://et2cloudprod/blink_rest_server/root.bayes_team/test_0812/chk-61431","discarded":true}]}</Checkpoints>

`JSON` 格式

{
 "RequestId": "A0623FAD-D4D7-4DFA-B126-D730C84338ED",
 "Checkpoints": "{\"counts\":{\"restored\":1,\"total\":61440,\"in_progress\":0,\"completed\":61440,\"failed\":0},\"summary\":{\"state_size\":{\"min\":4384,\"max\":4384,\"avg\":4384},\"end_to_end_duration\":{\"min\":9,\"max\":981,\"avg\":15},\"alignment_buffered\":{\"min\":0,\"max\":0,\"avg\":0}},\"latest\":{\"completed\":{\"@class\":\"completed\",\"id\":61440,\"status\":\"COMPLETED\",\"is_savepoint\":false,\"trigger_timestamp\":1610505450869,\"latest_ack_timestamp\":1610505450883,\"state_size\":4384,\"full_state_size\":4384,\"end_to_end_duration\":14,\"alignment_buffered\":0,\"num_subtasks\":1,\"num_acknowledged_subtasks\":1,\"tasks\":{},\"external_path\":\"hdfs://et2cloudprod/blink_rest_server/root.bayes_team/test_0812/chk-61440\",\"discarded\":false},\"savepoint\":null,\"failed\":null,\"restored\":{\"id\":7965,\"restore_timestamp\":1600879950869,\"is_savepoint\":false,\"external_path\":\"hdfs://

```
et2cloudprod/blink_rest_server/root.bayes_team/test_0812/chk-7965\"}},\"history\":[{\"@clas
s\":\"completed\",\"id\":61440,\"status\":\"COMPLETED\",\"is_savepoint\":false,\"trigger_ti
mestamp\":1610505450869,\"latest_ack_timestamp\":1610505450883,\"state_size\":4384,\"full_s
tate_size\":4384,\"end_to_end_duration\":14,\"alignment_buffered\":0,\"num_subtasks\":1,\"n
um_acknowledged_subtasks\":1,\"tasks\":{},\"external_path\":\"hdfs://et2cloudprod/blink_res
t_server/root.bayes_team/test_0812/chk-61440\",\"discarded\":false},{\"@class\":\"completed
\",\"id\":61439,\"status\":\"COMPLETED\",\"is_savepoint\":false,\"trigger_timestamp\":16105
05270869,\"latest_ack_timestamp\":1610505270888,\"state_size\":4384,\"full_state_size\":438
4,\"end_to_end_duration\":19,\"alignment_buffered\":0,\"num_subtasks\":1,\"num_acknowledged
_subtasks\":1,\"tasks\":{},\"external_path\":\"hdfs://et2cloudprod/blink_rest_server/root.b
ayes_team/test_0812/chk-61439\",\"discarded\":true},{\"@class\":\"completed\",\"id\":61438,
\"status\":\"COMPLETED\",\"is_savepoint\":false,\"trigger_timestamp\":1610505090869,\"lates
t_ack_timestamp\":1610505090882,\"state_size\":4384,\"full_state_size\":4384,\"end_to_end_d
uration\":13,\"alignment_buffered\":0,\"num_subtasks\":1,\"num_acknowledged_subtasks\":1,\"
tasks\":{},\"external_path\":\"hdfs://et2cloudprod/blink_rest_server/root.bayes_team/test_0
812/chk-61438\",\"discarded\":true},{\"@class\":\"completed\",\"id\":61437,\"status\":\"COM
PLETED\",\"is_savepoint\":false,\"trigger_timestamp\":1610504910869,\"latest_ack_timestamp\
":1610504910882,\"state_size\":4384,\"full_state_size\":4384,\"end_to_end_duration\":13,\"a
lignment_buffered\":0,\"num_subtasks\":1,\"num_acknowledged_subtasks\":1,\"tasks\":{},\"ext
ernal_path\":\"hdfs://et2cloudprod/blink_rest_server/root.bayes_team/test_0812/chk-61437\",
\"discarded\":true},{\"@class\":\"completed\",\"id\":61436,\"status\":\"COMPLETED\",\"is_sa
vepoint\":false,\"trigger_timestamp\":1610504730869,\"latest_ack_timestamp\":1610504730882,
\"state_size\":4384,\"full_state_size\":4384,\"end_to_end_duration\":13,\"alignment_buffere
d\":0,\"num_subtasks\":1,\"num_acknowledged_subtasks\":1,\"tasks\":{},\"external_path\":\"h
dfs://et2cloudprod/blink_rest_server/root.bayes_team/test_0812/chk-61436\",\"discarded\":tr
ue},{\"@class\":\"completed\",\"id\":61435,\"status\":\"COMPLETED\",\"is_savepoint\":false,
\"trigger_timestamp\":1610504550869,\"latest_ack_timestamp\":1610504550883,\"state_size\":4
384,\"full_state_size\":4384,\"end_to_end_duration\":14,\"alignment_buffered\":0,\"num_subt
asks\":1,\"num_acknowledged_subtasks\":1,\"tasks\":{},\"external_path\":\"hdfs://et2cloudpr
od/blink_rest_server/root.bayes_team/test_0812/chk-61435\",\"discarded\":true},{\"@class\":
\"completed\",\"id\":61434,\"status\":\"COMPLETED\",\"is_savepoint\":false,\"trigger_timest
amp\":1610504370869,\"latest_ack_timestamp\":1610504370885,\"state_size\":4384,\"full_state
_size\":4384,\"end_to_end_duration\":16,\"alignment_buffered\":0,\"num_subtasks\":1,\"num_a
cknowledged_subtasks\":1,\"tasks\":{},\"external_path\":\"hdfs://et2cloudprod/blink_rest_se
rver/root.bayes_team/test_0812/chk-61434\",\"discarded\":true},{\"@class\":\"completed\",\"
id\":61433,\"status\":\"COMPLETED\",\"is_savepoint\":false,\"trigger_timestamp\":1610504190
869,\"latest_ack_timestamp\":1610504190882,\"state_size\":4384,\"full_state_size\":4384,\"e
nd_to_end_duration\":13,\"alignment_buffered\":0,\"num_subtasks\":1,\"num_acknowledged_subt
asks\":1,\"tasks\":{},\"external_path\":\"hdfs://et2cloudprod/blink_rest_server/root.bayes_
team/test_0812/chk-61433\",\"discarded\":true},{\"@class\":\"completed\",\"id\":61432,\"sta
tus\":\"COMPLETED\",\"is_savepoint\":false,\"trigger_timestamp\":1610504010869,\"latest_ack
_timestamp\":1610504010896,\"state_size\":4384,\"full_state_size\":4384,\"end_to_end_durati
on\":27,\"alignment_buffered\":0,\"num_subtasks\":1,\"num_acknowledged_subtasks\":1,\"tasks
\":{},\"external_path\":\"hdfs://et2cloudprod/blink_rest_server/root.bayes_team/test_0812/c
hk-61432\",\"discarded\":true},{\"@class\":\"completed\",\"id\":61431,\"status\":\"COMPLETE
D\",\"is_savepoint\":false,\"trigger_timestamp\":1610503830869,\"latest_ack_timestamp\":161
0503830890,\"state_size\":4384,\"full_state_size\":4384,\"end_to_end_duration\":21,\"alignm
ent_buffered\":0,\"num_subtasks\":1,\"num_acknowledged_subtasks\":1,\"tasks\":{},\"external
_path\":\"hdfs://et2cloudprod/blink_rest_server/root.bayes_team/test_0812/chk-61431\",\"dis
carded\":true}]}"
}
```

5.5. GetInstanceConfig

您可以通过GetInstanceConfig OpenAPI获取作业实例运行时的具体参数，非运行状态的作业调用此API会返回错误。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
GET /api/v2/projects/[projectName]/jobs/[jobName]/instances/[instanceId]/config HTTP/1.1
```

请求参数

名称	类型	位置	是否必选	示例值	描述
instanceId	Long	Path	是	512532	实例ID： • 流作业：填写实际运行的实例ID或直接填写-1（当前运行的实例）。 • 批作业：可以通过ListInstance和StartJob等接口获得。
jobName	String	Path	是	test_0812	作业名称。
projectName	String	Path	是	bayes_team	项目名称。
RegionId	String	Header	是	cn-hangzhou	区域ID。

返回数据

名称	类型	示例值	描述
RequestId	String	DBBB9DE4-97F5-4BB4-9931-16FE88603D36	请求ID
		{\"jid\":\"166845b9753a86282144c59d0c3d82b7\",\"name\":\"test_0812\",\"exe	

名称	类型	示例值	描述
		cution-config\":{\"execution-mode\":\"PIPELINED\",\"restart-strategy\":\"Cluster level default restart strategy\",\"job-parallelism\":-1,\"object-reuse-mode\":true,\"user-config\":{\"env.java.opts.jobmanager\":\"-Xloggc:<LOG_DIR>/jobmanager-gc.log -XX:+UseGCLogFileRotation -XX:NumberOfGCLogFiles=2 -XX:GCLogFileSize=512M\",\"env.java.opts\":\"-verbose:gc -XX:NewRatio=3 -XX:+PrintGCDetails -XX:+PrintGCDateStamps -XX:ParallelGCThreads=4\",\"metrics.reporter.kmonitor.class\":\"com.alibaba.blink.monitor.RichKmonitorReporter\",\"sharedslot-enabled.default\":\"false\",\"job.app-master-core\":\"0.25\",\"high-availability.zookeeper.client.session-timeout\":\"120000\",\"yarn.container-launcher-number\":\"100\",\"resource.cpu.scale-up.ratio\":\"1.5\",\"resource.cpu.scale-down.ratio\":\"1.5\",\"high-availability.storageDir\":\"hdfs://et2cloudprod/tmp/blink3.0/ha\",\"state.backend.niagara.ttl.ms\":\"129600000\",\"st	

名称	类型	示例值	描述
Config	String	ate.backend.niagara.checkpointpath\":\"hdfs://et2cloudprod/blink_rest_server/root.bayes_team/test_0812/\",\"taskmanager.network.memory.buffers-per-external-blocking-channel\":\"16\",\"__inner__queueName__\":\"root.bayes_team\",\"metrics.reporters\":\"kmonitor,jmx\",\"metrics.reporter.kmonitor.interval\":\"10 SECONDS\",\"healthmonitor.high-cpu-detector.severe.threashold\":\"0.9\",\"yarn.per-job-cluster.include-user-jar\":\"LAST\",\"parallelism.scale.ratio.max\":\"4\",\"blink.job.source.startTime\":\"2020-09-07 10:34:05\",\"stsAccesskey\":\"v91V042KRCL7mFEYflP02INwSv71m0Jz18RUsNbRzFM=\",\"healthmonitor.high-cpu-detector.threashold\":\"0.9\",\"state.backend.local-recovery\":\"true\",\"__inner__clusterName__\":\"et2cloudprod\",\"high-availability.zookeeper.quorum\":\"zk.et2cloudprod.blink.alibaba-inc.com:12181\",\"restart-strategy.fixed-delay.delay\":\"10s\",\"metrics.process-tree.tm.enabled\":\"true\",\"metrics.process-tree.tm.update-interval\":\"10000\"	作业运行的相关参数

名称	类型	示例值	描述
		,\"metrics.reporter.jmx.class\":\"org.apache.flink.metrics.jmx.JMXReporter\",\"slot.enable-shared-slot\":\"false\",\"yarn.vcore-ratio\":\"100\",\"blink.checkpoint.external.cleanup\":\"RETAIN_ON_CANCELLATION\",\"healthmanager.enabled\":\"true\",\"metrics.reporter.jmx.port\":\"1025-10000\",\"taskmanager.network.request-backoff.max\":\"300000\",\"akka.watch.heartbeat.interval\":\"10s\",\"parallelism.scale.ratio.min\":\"2\",\"blink.job.timeZone\":\"Asia/Shanghai\",\"metrics.reporter.kmonitor.tag.value.length\":\"64\",\"yarn.application-attempts\":\"100\",\"__inner_zk_quorum__\":\"zk.et2cloudprod.blink.alibaba-inc.com:12181\",\"akka.ask.timeout\":\"120s\",\"stsAccessId\":\"9juusi0kMFGpRs8c\",\"task.external.shuffle.max-concurrent-requests\":\"512\",\"blink.job.checkpoint.dir\":\"hdfs://et2cloudprod/blink_rest_server/root.bayes_team/test_0812/\",\"blink.job.source.startTime.mills\":\"1599446045306\",\"blink.checkpoint.interval.ms\":\"180000\",\"taskmanager.network.memory.floating-buffers-per-	

名称	类型	示例值	描述
		gate\":\"256\",\"__inner__projectName__\":\"bayes_team\",\"web.timeout\":\"120000\",\"jobmanager.heap.mb\":\"1024\",\"action.selector.rescale.resource.first\":\"true\",\"statebackend.onheap\":\"false\",\"restart-strategy.fixed-delay.attempts\":\"2147483647\",\"stsUid\":\"1709064687573327\",\"jobmanager.execution.attempts-history-size\":\"100\",\"slotmanager.taskmanager-timeout\":\"600000\",\"task.external.shuffle.compression.enable\":\"true\",\"__inner__jobName__\":\"test_0812\",\"akka.framesize\":\"102400kB\",\"state.checkpoints.create-subdirs\":\"false\",\"high-availability\":\"zookeeper\",\"blink.checkpoint.path\":\"hdfs://et2cloudprod/blink_rest_server/root.bayes_team/test_0812/\",\"stsRegionId\":\"cn-shanghai\",\"blink.checkpoint.mode\":\"EXACTLY_ONCE\",\"restart-strategy\":\"fixed-delay\",\"blink.checkpoint.external.path\":\"hdfs://et2cloudprod/blink_rest_server/root.bayes_team/test_0812/external_cp\",\"env.java.opts.taskmanager\":\"-Xloggc:<LOG_DIR>/taskmanager-gc.log -	

名称	类型	示例值	描述
		XX:+UseGCLogFileRotation - XX:NumberOfGCLogFiles=2 - XX:GCLogFileSize=512M\"}}}	

示例

请求示例

```
/api/v2/projects/bayes_team/jobs/test_0812/instances/512532/config
```

正常返回示例

`XML` 格式

<RequestId>DBBB9DE4-97F5-4BB4-9931-16FE88603D36</RequestId>
<Config>{"jid":"166845b9753a86282144c59d0c3d82b7","name":"test_0812","execution-config":{"execution-mode":"PIPELINED","restart-strategy":"Cluster level default restart strategy","job-parallelism":-1,"object-reuse-mode":true,"user-config":{"env.java.opts.jobmanager":"-Xloggc:<LOG_DIR>/jobmanager-gc.log -XX:+UseGCLogFileRotation -XX:NumberOfGCLogFiles=2 -XX:GCLogFileSize=512M","env.java.opts":"-verbose:gc -XX:NewRatio=3 -XX:+PrintGCDetails -XX:+PrintGCDateStamps -XX:ParallelGCThreads=4","metrics.reporter.kmonitor.class":"com.alibaba.blink.monitor.RichKmonitorReporter","sharedslot-enabled.default":"false","job.app-master-core":"0.25","high-availability.zookeeper.client.session-timeout":"120000","yarn.container-launcher-number":"100","resource.cpu.scale-up.ratio":"1.5","resource.cpu.scale-down.ratio":"1.5","high-availability.storageDir":"hdfs://et2cloudprod/tmp/blink3.0/ha","state.backend.niagara.ttl.ms":"129600000","state.backend.niagara.checkpointpath":"hdfs://et2cloudprod/blink_rest_server/root.bayes_team/test_0812/","taskmanager.network.memory.buffers-per-external-blocking-channel":"16","__inner__queueName__":"root.bayes_team","metrics.reporters":"kmonitor,jmx","metrics.reporter.kmonitor.interval":"10 SECONDS","healthmonitor.high-cpu-detector.severe.threashold":"0.9","yarn.per-job-cluster.include-user-jar":"LAST","parallelism.scale.ratio.max":"4","blink.job.source.startTime":"2020-09-07 10:34:05","stsAccesskey":"v91V042KRCL7mFEYflP02INwSv71m0Jz18RUsNbRzFM=","healthmonitor.high-cpu-detector.threashold":"0.9","state.backend.local-recovery":"true","__inner__clusterName__":"et2cloudprod","high-availability.zookeeper.quorum":"zk.et2cloudprod.blink.alibaba-inc.com:12181","restart-strategy.fixed-delay.delay":"10 s","metrics.process-tree.tm.enabled":"true","metrics.process-tree.tm.update-interval":"10000","metrics.reporter.jmx.class":"org.apache.flink.metrics.jmx.JMXReporter","slot.enable-shared-slot":"false","yarn.vcore-ratio":"100","blink.checkpoint.external.cleanup":"RETAIN_ON_CANCELLATION","healthmanager.enabled":"true","metrics.reporter.jmx.port":"1025-10000","taskmanager.network.request-backoff.max":"300000","akka.watch.heartbeat.interval":"10 s","parallelism.scale.ratio.min":"2","blink.job.timeZone":"Asia/Shanghai","metrics.reporter.kmonitor.tag.value.length":"64","yarn.application-attempts":"100","__inner_zk_quorum__":"zk.et2cloudprod.blink.alibaba-inc.com:12181","akka.ask.timeout":"120 s","stsAccessId":"9juusi0kMFGpRs8c","task.external.shuffle.max-concurrent-requests":"512","blink.job.checkpoint.dir":"hdfs://et2cloudprod/blink_rest_server/root.bayes_team/test_0812/","blink.job.source.startTime.mills":"1599446045306","blink.checkpoint.interval.ms":"180000","taskmanager.network.memory.floating-buffers-per-gate":"256","__inner__projectName__":"bayes_team","web.timeout":"120000","jobmanager.heap.mb":"1024","action.selector.rescale.resource.first":"true","statebackend.onheap":"false","restart-strategy.fixed-delay.attempts":"2147483647","stsUid":"1709064687573327","jobmanager.execution.attempts-history-size":"100","slotmanager.taskmanager-timeout":"600000","task.external.shuffle.compression.enable":"true","__inner__jobName__":"test_0812","akka.framesize":"102400kB","state.checkpoints.create-subdirs":"false","high-availability":"zookeeper","blink.checkpoint.path":"hdfs://et2cloudprod/blink_rest_server/root.bayes_team/test_0812/","stsRegionId":"cn-shanghai","blink.checkpoint.mode":"EXACTLY_ONCE","restart-strategy":"fixed-delay","blink.checkpoint.external.path":"hdfs://et2cloudprod/blink_rest_server/root.bayes_team/test_0812/external_cp","env.java.opts.taskmanager":"-Xloggc:<LOG_DIR>/taskmanager-gc.log -XX:+UseGCLogFileRotation -XX:NumberOfGCLogFiles=2 -XX:GCLogFileSize=512M"}}}</Config>

`JSON` 格式

```
{
 "RequestId": "DBBB9DE4-97F5-4BB4-9931-16FE88603D36",
 "Config": "{\"jid\":\"166845b9753a86282144c59d0c3d82b7\",\"name\":\"test_0812\",\"executio
n-config\":{\"execution-mode\":\"PIPELINED\",\"restart-strategy\":\"Cluster level default r
estart strategy\",\"job-parallelism\":-1,\"object-reuse-mode\":true,\"user-config\":{\"env.
java.opts.jobmanager\":\"-Xloggc:<LOG_DIR>/jobmanager-gc.log -XX:+UseGCLogFileRotation -XX:
NumberOfGCLogFiles=2 -XX:GCLogFileSize=512M\",\"env.java.opts\":\"-verbose:gc -XX:NewRatio=
3 -XX:+PrintGCDetails -XX:+PrintGCDateStamps -XX:ParallelGCThreads=4\",\"metrics.reporter.k
monitor.class\":\"com.alibaba.blink.monitor.RichKmonitorReporter\",\"sharedslot-enabled.def
ault\":\"false\",\"job.app-master-core\":\"0.25\",\"high-availability.zookeeper.client.sess
ion-timeout\":\"120000\",\"yarn.container-launcher-number\":\"100\",\"resource.cpu.scale-up
.ratio\":\"1.5\",\"resource.cpu.scale-down.ratio\":\"1.5\",\"high-availability.storageDir\"
:\"hdfs://et2cloudprod/tmp/blink3.0/ha\",\"state.backend.niagara.ttl.ms\":\"129600000\",\"s
tate.backend.niagara.checkpointpath\":\"hdfs://et2cloudprod/blink_rest_server/root.bayes_te
am/test_0812/\",\"taskmanager.network.memory.buffers-per-external-blocking-channel\":\"16\"
,\"__inner__queueName__\":\"root.bayes_team\",\"metrics.reporters\":\"kmonitor,jmx\",\"metr
ics.reporter.kmonitor.interval\":\"10 SECONDS\",\"healthmonitor.high-cpu-detector.severe.th
reashold\":\"0.9\",\"yarn.per-job-cluster.include-user-jar\":\"LAST\",\"parallelism.scale.r
atio.max\":\"4\",\"blink.job.source.startTime\":\"2020-09-07 10:34:05\",\"stsAccesskey\":\"
v91V042KRCL7mFEYflP02INwSv71m0Jz18RUsNbRzFM=\",\"healthmonitor.high-cpu-detector.threashold
\":\"0.9\",\"state.backend.local-recovery\":\"true\",\"__inner__clusterName__\":\"et2cloudp
rod\",\"high-availability.zookeeper.quorum\":\"zk.et2cloudprod.blink.alibaba-inc.com:12181\
",\"restart-strategy.fixed-delay.delay\":\"10 s\",\"metrics.process-tree.tm.enabled\":\"tru
e\",\"metrics.process-tree.tm.update-interval\":\"10000\",\"metrics.reporter.jmx.class\":\"
org.apache.flink.metrics.jmx.JMXReporter\",\"slot.enable-shared-slot\":\"false\",\"yarn.vco
re-ratio\":\"100\",\"blink.checkpoint.external.cleanup\":\"RETAIN_ON_CANCELLATION\",\"healt
hmanager.enabled\":\"true\",\"metrics.reporter.jmx.port\":\"1025-10000\",\"taskmanager.netw
ork.request-backoff.max\":\"300000\",\"akka.watch.heartbeat.interval\":\"10 s\",\"paralleli
sm.scale.ratio.min\":\"2\",\"blink.job.timeZone\":\"Asia/Shanghai\",\"metrics.reporter.kmon
itor.tag.value.length\":\"64\",\"yarn.application-attempts\":\"100\",\"__inner_zk_quorum__\
":\"zk.et2cloudprod.blink.alibaba-inc.com:12181\",\"akka.ask.timeout\":\"120 s\",\"stsAcces
sId\":\"9juusi0kMFGpRs8c\",\"task.external.shuffle.max-concurrent-requests\":\"512\",\"blin
k.job.checkpoint.dir\":\"hdfs://et2cloudprod/blink_rest_server/root.bayes_team/test_0812/\"
,\"blink.job.source.startTime.mills\":\"1599446045306\",\"blink.checkpoint.interval.ms\":\"
180000\",\"taskmanager.network.memory.floating-buffers-per-gate\":\"256\",\"__inner__projec
tName__\":\"bayes_team\",\"web.timeout\":\"120000\",\"jobmanager.heap.mb\":\"1024\",\"actio
n.selector.rescale.resource.first\":\"true\",\"statebackend.onheap\":\"false\",\"restart-st
rategy.fixed-delay.attempts\":\"2147483647\",\"stsUid\":\"1709064687573327\",\"jobmanager.e
xecution.attempts-history-size\":\"100\",\"slotmanager.taskmanager-timeout\":\"600000\",\"t
ask.external.shuffle.compression.enable\":\"true\",\"__inner__jobName__\":\"test_0812\",\"a
kka.framesize\":\"102400kB\",\"state.checkpoints.create-subdirs\":\"false\",\"high-availabi
lity\":\"zookeeper\",\"blink.checkpoint.path\":\"hdfs://et2cloudprod/blink_rest_server/root
.bayes_team/test_0812/\",\"stsRegionId\":\"cn-shanghai\",\"blink.checkpoint.mode\":\"EXACTL
Y_ONCE\",\"restart-strategy\":\"fixed-delay\",\"blink.checkpoint.external.path\":\"hdfs://e
t2cloudprod/blink_rest_server/root.bayes_team/test_0812/external_cp\",\"env.java.opts.taskm
anager\":\"-Xloggc:<LOG_DIR>/taskmanager-gc.log -XX:+UseGCLogFileRotation -XX:NumberOfGCLog
Files=2 -XX:GCLogFileSize=512M\"}}}"
}
```


5.6. GetInstanceDetail

您可以通过GetInstanceDetail OpenAPI获取实例运行的DAG图，以JSON的形式返回。非运行状态的作业调用此API会返回错误。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
GET /api/v2/projects/[projectName]/jobs/[jobName]/instances/[instanceId]/details HTTP/1.1
```

请求参数

名称	类型	位置	是否必选	示例值	描述
instanceId	Long	Path	是	403936	实例ID： • 流作业：填写实际运行的实例ID或直接填写-1（当前运行的实例）。 • 批作业：可以通过ListInstance和StartJob等接口获得。
jobName	String	Path	是	test_0812	作业名称。
projectName	String	Path	是	bayes_team	项目名称。
RegionId	String	Header	是	cn-hangzhou	区域ID。

返回数据

名称	类型	示例值	描述
		{\"jid\":\"166845b9753a86282144c59d0c3d82b7\",\"name\":\"test_0812\",\"isStoppable\":false,\"state\":\"CANCELED\",\"start-time\":1600879950827,\"end-time\":1610505455432,\"duration\":962	

名称	类型	示例值	描述
		5504605,\"now\":1610505455494,\"timestamps\":{\"CREATED\":1600879950827,\"RUNNING\":1600879950869,\"RECONCILING\":0,\"FAILING\":0,\"FINISHED\":0,\"CANCELLING\":1610505454138,\"RESTARTING\":1600879940762,\"SUSPENDING\":0,\"SUSPENDED\":0,\"FAILED\":0,\"CANCELED\":1610505455432},\"vertices\":[{\"id\":\"717c7b8afebbfb7137f6f0f99beb2a94\",\"topology-id\":0,\"name\":\"Source: RandomSource-test_input-Stream -> SourceConversion(table:[builtin, default, _DataStreamTable_0, source: [RandomSource-test_input]], fields: (s, rowtime)) -> WatermarkAssigner(rowtime: rowtime, offset: 4000) -> Calc(select: (s AS c)) -> SinkConversion to Tuple2 -> Flat Map -> Sink: Print to System.out\",\"parallelism\":1,\"status\":\"CANCELED\",\"start-time\":1600880000174,\"end-time\":1610505455427,\"duration\":9625455253,\"tasks\":{\"CANCELING\":0,\"RUNNING\":0,\"SCHEDULED\":0,\"RECONCILING\":0,\"DEPLOYING\":0,\"CREATED\":0,\"CANCELED\":1,\"FINISHED\":0,\"FAIL	

名称	类型	示例值	描述
Detail	String	ED\":0},\"metrics\":{\"read-bytes\":0,\"read-bytes-complete\":true,\"write-bytes\":0,\"write-bytes-complete\":true,\"read-records\":0,\"read-records-complete\":true,\"write-records\":0,\"write-records-complete\":true,\"buffers-in-pool-usage-max\":0.0,\"buffers-in-pool-usage-max-complete\":true,\"buffers-out-pool-usage-max\":0.0,\"buffers-out-pool-usage-max-complete\":true,\"tps\":-1.0,\"tps-complete\":true,\"delay\":-1,\"delay-complete\":true}}],\"status-counts\":{\"CANCELING\":0,\"RUNNING\":0,\"SCHEDULED\":0,\"RECONCILING\":0,\"DEPLOYING\":0,\"CREATED\":0,\"CANCELED\":1,\"FINISHED\":0,\"FAILED\":0},\"plan\":{\"jid\":\"166845b9753a86282144c59d0c3d82b7\",\"name\":\"test_0812\",\"nodes\":[{\"id\":\"717c7b8afebbfb7137f6f0f99beb2a94\",\"parallelism\":1,\"operator\":\"\",\"operator_strategy\":\"\",\"description\":\"Source: RandomSource-test_input-Stream -	作业运行的DAG信息

名称	类型	示例值	描述
		> SourceConversion(table:[builtin, default, _DataStreamTable_0, source: [RandomSource-test_input]], fields: (s, rowtime)) -> WatermarkAssigner(rowtime: rowtime, offset: 4000) -> Calc(select: (s AS c)) -> SinkConversion to Tuple2 -> Flat Map -> Sink: Print to System.out\",\"optimizer_properties\":{}}}	
RequestId	String	2FC99DE0-9E70-4E57-9B16-5CBDB10211FB	请求ID。

示例

请求示例

```
/api/v2/projects/bayes_team/jobs/test_0812/instances/412532/details
```

正常返回示例

`XML` 格式

<RequestId>2FC99DE0-9E70-4E57-9B16-5CBDB10211FB</RequestId>
<Detail>{"jid":"166845b9753a86282144c59d0c3d82b7","name":"test_0812","isStoppable":false,"state":"CANCELED","start-time":1600879950827,"end-time":1610505455432,"duration":9625504605,"now":1610505455494,"timestamps":{"CREATED":1600879950827,"RUNNING":1600879950869,"RECONCILING":0,"FAILING":0,"FINISHED":0,"CANCELLING":1610505454138,"RESTARTING":1600879940762,"SUSPENDING":0,"SUSPENDED":0,"FAILED":0,"CANCELED":1610505455432},"vertices":[{"id":"717c7b8afebbfb7137f6f0f99beb2a94","topology-id":0,"name":"Source: RandomSource-test_input-Stream -> SourceConversion(table:[builtin, default, _DataStreamTable_0, source: [RandomSource-test_input]], fields:(s, rowtime)) -> WatermarkAssigner(rowtime: rowtime, offset: 4000) -> Calc(select: (s AS c)) -> SinkConversion to Tuple2 -> Flat Map -> Sink: Print to System.out","parallelism":1,"status":"CANCELED","start-time":1600880000174,"end-time":1610505455427,"duration":9625455253,"tasks":{"CANCELING":0,"RUNNING":0,"SCHEDULED":0,"RECONCILING":0,"DEPLOYING":0,"CREATED":0,"CANCELED":1,"FINISHED":0,"FAILED":0},"metrics":{"read-bytes":0,"read-bytes-complete":true,"write-bytes":0,"write-bytes-complete":true,"read-records":0,"read-records-complete":true,"write-records":0,"write-records-complete":true,"buffers-in-pool-usage-max":0.0,"buffers-in-pool-usage-max-complete":true,"buffers-out-pool-usage-max":0.0,"buffers-out-pool-usage-max-complete":true,"tps":-1.0,"tps-complete":true,"delay":-1,"delay-complete":true}}],"status-counts":{"CANCELING":0,"RUNNING":0,"SCHEDULED":0,"RECONCILING":0,"DEPLOYING":0,"CREATED":0,"CANCELED":1,"FINISHED":0,"FAILED":0},"plan":{"jid":"166845b9753a86282144c59d0c3d82b7","name":"test_0812","nodes":[{"id":"717c7b8afebbfb7137f6f0f99beb2a94","parallelism":1,"operator":"","operator_strategy":"","description":"Source: RandomSource-test_input-Stream -> SourceConversion(table:[builtin, default, _DataStreamTable_0, source: [RandomSource-test_input]], fields:(s, rowtime)) -> WatermarkAssigner(rowtime: rowtime, offset: 4000) -> Calc(select: (s AS c)) -> SinkConversion to Tuple2 -> Flat Map -> Sink: Print to System.out","optimizer_properties":{}}]}}</Detail>

`JSON` 格式

```
{
 "RequestId": "2FC99DE0-9E70-4E57-9B16-5CBDB10211FB",
 "Detail": "{\"jid\":\"166845b9753a86282144c59d0c3d82b7\",\"name\":\"test_0812\",\"isStoppable\":false,\"state\":\"CANCELED\",\"start-time\":1600879950827,\"end-time\":1610505455432,\"duration\":9625504605,\"now\":1610505455494,\"timestamps\":{\"CREATED\":1600879950827,\"RUNNING\":1600879950869,\"RECONCILING\":0,\"FAILING\":0,\"FINISHED\":0,\"CANCELLING\":1610505454138,\"RESTARTING\":1600879940762,\"SUSPENDING\":0,\"SUSPENDED\":0,\"FAILED\":0,\"CANCELED\":1610505455432},\"vertices\":[{\"id\":\"717c7b8afebbfb7137f6f0f99beb2a94\",\"topology-id\":0,\"name\":\"Source: RandomSource-test_input-Stream -> SourceConversion(table:[builtin, default, _DataStreamTable_0, source: [RandomSource-test_input]], fields:(s, rowtime)) -> WatermarkAssigner(rowtime: rowtime, offset: 4000) -> Calc(select: (s AS c)) -> SinkConversion to Tuple2 -> Flat Map -> Sink: Print to System.out\",\"parallelism\":1,\"status\":\"CANCELED\",\"start-time\":1600880000174,\"end-time\":1610505455427,\"duration\":9625455253,\"tasks\":{\"CANCELING\":0,\"RUNNING\":0,\"SCHEDULED\":0,\"RECONCILING\":0,\"DEPLOYING\":0,\"CREATED\":0,\"CANCELED\":1,\"FINISHED\":0,\"FAILED\":0},\"metrics\":{\"read-bytes\":0,\"read-bytes-complete\":true,\"write-bytes\":0,\"write-bytes-complete\":true,\"read-records\":0,\"read-records-complete\":true,\"write-records\":0,\"write-records-complete\":true,\"buffers-in-pool-usage-max\":0.0,\"buffers-in-pool-usage-max-complete\":true,\"buffers-out-pool-usage-max\":0.0,\"buffers-out-pool-usage-max-complete\":true,\"tps\":-1.0,\"tps-complete\":true,\"delay\":-1,\"delay-complete\":true}}],\"status-counts\":{\"CANCELING\":0,\"RUNNING\":0,\"SCHEDULED\":0,\"RECONCILING\":0,\"DEPLOYING\":0,\"CREATED\":0,\"CANCELED\":1,\"FINISHED\":0,\"FAILED\":0},\"plan\":{\"jid\":\"166845b9753a86282144c59d0c3d82b7\",\"name\":\"test_0812\",\"nodes\":[{\"id\":\"717c7b8afebbfb7137f6f0f99beb2a94\",\"parallelism\":1,\"operator\":\"\",\"operator_strategy\":\"\",\"description\":\"Source: RandomSource-test_input-Stream -&gt; SourceConversion(table:[builtin, default, _DataStreamTable_0, source: [RandomSource-test_input]], fields:(s, rowtime)) -&gt; WatermarkAssigner(rowtime: rowtime, offset: 4000) -&gt; Calc(select: (s AS c)) -&gt; SinkConversion to Tuple2 -&gt; Flat Map -&gt; Sink: Print to System.out\",\"optimizer_properties\":{}}]}}"
}
```


5.7. GetInstanceExceptions

您可以通过GetInstanceExceptions OpenAPI获取运行实例的Failover信息。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
GET /api/v2/projects/[projectName]/jobs/[jobName]/instances/[instanceId]/exceptions HTTP/1.1
```

请求参数

名称	类型	位置	是否必选	示例值	描述
instanceId	Long	Path	是	403936	实例ID： • 流作业：填写实际运行的实例ID或直接填写-1（当前运行的实例）。 • 批作业：可以通过ListInstance、StartJob等接口获得。
jobName	String	Path	是	test_0812	作业名称。
projectName	String	Path	是	bayes_team	项目名称。
RegionId	String	Header	是	cn-hangzhou	区域ID。

返回数据

名称	类型	示例值	描述
Exceptions	String	{\"root-exception\":null,\"timestamp\":null,\"all-exceptions\":[],\"truncated\":false}	Failover的具体信息。
RequestId	String	97381B4E-EE09-497A-A4AB-9AA6E5B7D0DE	请求ID。

示例

请求示例

```
/api/v2/projects/project1/jobs/job1/instances/412532/exceptions
```

正常返回示例

`XML` 格式

```
<RequestId>97381B4E-EE09-497A-A4AB-9AA6E5B7D0DE</RequestId>
<Exceptions>{"root-exception":null,"timestamp":null,"all-exceptions":[],"truncated":false}</Exceptions>
```

`JSON` 格式

```
{
 "RequestId": "97381B4E-EE09-497A-A4AB-9AA6E5B7D0DE",
 "Exceptions": "{\"root-exception\":null,\"timestamp\":null,\"all-exceptions\":[],\"truncated\":false}"
}
```

5.8. GetInstanceFinalState

您可以通过GetInstanceFinalState OpenAPI查看运行实例在YARN上的最终状态。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
GET /api/v2/projects/[projectName]/jobs/[jobName]/instances/[instanceId]/finalstate HTTP/1.1
```

请求参数

名称	类型	位置	是否必选	示例值	描述
instanceId	Long	Path	是	412532	实例ID： • 流作业：填写实际运行的实例ID或直接填写-1（当前运行的实例）。 • 批作业：可以通过ListInstance或StartJob等接口获得。
jobName	String	Path	是	job1	作业名称。
projectName	String	Path	是	project1	项目名称。
RegionId	String	Header	是	cn-hangzhou	区域ID。

返回数据

名称	类型	示例值	描述

名称	类型	示例值	描述
Finalstate	String	RUNNING	YARN上的最终状态。
RequestId	String	3F4ADBCF-B116-4680-80B3-E9A139841A0D	请求ID。

示例

请求示例

```
/api/v2/projects/project1/jobs/job1/instances/412532/finalstate
```

正常返回示例

`XML` 格式

```
<requestId>3F4ADBCF-B116-4680-80B3-E9A139841A0D</requestId>
```

`JSON` 格式

```
{
    "requestId": "3F4ADBCF-B116-4680-80B3-E9A139841A0D"
}
```

5.9. GetInstanceMetric

您可以通过GetInstanceMetric获取正在运行实例的所有Metric信息。

> **说明** 使用GetInstanceMetric调用未运行的实例的Metric时，会产生报错。metricJson参数的详细请参见opentsdb标准协议。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
POST /api/v2/projects/[projectName]/jobs/[jobName]/metric HTTP/1.1
```

请求参数

名称	类型	位置	是否必选	示例值	描述
jobName	String	Path	是	job1	作业名称。
metricJson	String	Body	是	{"start":1606291390000,"end":1606294626216,"limit":"avg:sample:50","queries":[{"downsample":"20s-avg","metric":"blink.projectName.jobName.task_failover.rate","granularity":"20s","aggregator":"max"}]}	使用特定JSON来获取Metric： • start：Metric开始时间。使用13位时间戳，精确到毫秒。 • limit：取各条线值类型：max、avg和min。取值：bottom、above和sample。 • end：Metric结束时间。使用13位时间戳，精确到毫秒。 • downsample：采样方式时间（秒）。取值：max、avg和min。 • metric：blink.项目名称.作业名称.指标名称。常用指标名称例如：业务延时（delay）。 • granularity：采样时间，每隔多少秒取一个点。需要根据start和end的进行计算。采集时间点太多会造成调用超时。 • aggregator：聚合方式。按照采样时间将底层的点聚合后输出为一个点。聚合方式有最大值max，最小值min和平均值avg。
projectName	String	Path	是	project1	项目名称。
instanceId	Long	Query	否	-1	InstanceID。流作业和批作业对应的实例ID如下： • 流作业：只有一个运行实例，此处填 `-1L` ，指在线上运行的。 • 批作业：可以通过 `ListInstance` 接口或者 `Startjob` 接口获得。
RegionId	String	Header	否	cn-hangzhou	地域ID。

返回数据

名称	类型	示例值	描述
Metrics	Array of Metric		Metric信息详情。

名称	类型	示例值	描述
Metric			
Dps	Map	k:v	时间点和对应的Metric值。
MetricName	String	blink.bayes_team.ss2.delay	Metric名称。
Summary	Float	494400448	聚合后的Metric值。
Tags	Map	k:v	Metric标记。
RequestId	String	4D7AB7CF-4EF2-4635-9137-51803CD71DAD	请求ID。

Metric名称对应表

参数名称	指标名称
业务延时（ms）	`delay`
数据滞留时间（ms）	`fetched_delay`
数据间隔时间（ms）	`no_data_delay`
作业失败率	`task_failover.rate`
Source的tps数据输入	`tps.rate`
Sink的数据输出	`sink.outTps.rate`
Source的RPS数据输出	`parserTps.rate`
Source的数据流量输入（byte）	`inBps.rate`
Source的脏数据比例	`parserSkipCount`
Checkpoint Duration（ms）	`lastCheckpointDuration`

参数名称	指标名称
Checkpoint大小（byte）	`lastCheckpointSize`
checkpoint对齐时间（ns）	`checkpointAlignmentTime`
checkpoint数量	`checkpoints`
获取state的时间（ns）	`rocksdb_get.mean`
写入state的时间（ns）	`rocksdb_put.mean`
seek（ns）	`niagara_seek.meam / rocksdb_seek.mean`
state大小（byte）	`state.state_size / state.state_newsize`
GMS GC Time（ms）	`Status.JVM.GarbageCollector.ConcurrentMarkSweep.Time`
GMS GC Rate	`Status.JVM.GarbageCollector.ConcurrentMarkSweep.Count`
WaterMark Delay（ms）	`watermarkLatency`
数据迟到丢弃TPS	`lateRecordsDroppedRate`
数据迟到累计丢弃数	`numLateRecordsDropped`
读TT延时（ms）	`input.tt.readLatency`
Task Input TPS	`numRecordsInPerSecond.rate`
Task Output TPS	`numRecordsOutPerSecond.rate`
Input Queue Usage	`buffers.inPoolUsage`
Output Queue Usage	`buffers.outPoolUsage`

参数名称	指标名称
Time Used In Processing Per Second（ns）	`taskLatency.sum`
Time Used In Waiting Oputput Per Second（ns）	`waitOutput.sum`
TaskLatency Histogram Mean（ns）	`taskLatency.histogram.mean`
WaitOutput Histogram Mean（ns）	`waitOutput.histogram.mean`
WaitInput Histogram Mean（ns）	`waitInput.histogram.mean`
PartitionLatency Mean（ns）	`partitionLatency.mean`
Process MEM Rss（kb）	`Status.JVM.Memory.Process.rss`
CPU Usage	`Status.JVM.CPU.Usage`
Memory Heap Used（byte）	`Status.JVM.Memory.Heap.Used`
Memory NonHeap Used（byte）	`Status.JVM.Memory.NonHeap.Used`
Threads Count	`Status.JVM.Threads.Count`
GC(CMS)	`Status.JVM.GarbageCollector.ConcurrentMarkSweep.Count`

示例

请求示例

```
/api/v2/projects/project1/jobs/job1/metric
{\"metricJson\":\"{\\"start\\":1606291390000,\\"end\\":1606294626216,\\"limit\\":\\"avg:sample:50\\",\\"queries\\":[{\\"downsample\\":\\"20s-avg\\",\\"metric\\":\\"blink.projectName.jobName.task_failover.rate\\",\\"granularity\\":\\"20s\\",\\"aggregator\\":\\"max\\"}]}\"}
```

正常返回示例

`XML` 格式

```
<requestId>4D7AB7CF-4EF2-4635-9137-51803CD71DAD</requestId>
<metrics>
    <metricName>blink.bayes_team.ss2.delay</metricName>
    <dps>
        <1574247000>4.929E+8</1574247000>
        <1574247600>4.9350016E+8</1574247600>
        <1574248200>494100352</1574248200>
        <1574248800>494700544</1574248800>
        <1574249400>495300736</1574249400>
        <1574250000>495900896</1574250000>
    </dps>
    <summary>494400448</summary>
    <tags/>
</metrics>
```

`JSON` 格式

```
{
    "requestId":"4D7AB7CF-4EF2-4635-9137-51803CD71DAD",
    "metrics":[
        {
            "metricName":"blink.bayes_team.ss2.delay",
            "dps":{
                "1574247000":"4.929E+8",
                "1574247600":"4.9350016E+8",
                "1574248200":"494100352",
                "1574248800":"494700544",
                "1574249400":"495300736",
                "1574250000":"495900896"
            },
            "summary":494400448,
            "tags":{}
        }
    ]
}
```

5.10. GetInstanceRunSummary

您可以通过GetInstanceRunSummary OpenAPI获取作业运行实例的运行信息。该OpenAPI是GetInstance接口的轻量级版本，会返回作业运行时的部分信息，相比GetInstance接口更加轻量级。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
GET /api/v2/projects/[projectName]/jobs/[jobName]/instances/[instanceId]/runsummary HTTP/1.
1
```

请求参数

名称	类型	位置	是否必选	示例值	描述
instanceId	Long	Path	是	-1	InstanceID： • 流作业：填写实际运行的实例ID或直接填写-1（当前运行的实例）。 • 批作业：可以通过ListInstance、StartJob等接口获得。
jobName	String	Path	是	job1	作业名称。
projectName	String	Path	是	project1	项目名称。
RegionId	String	Header	否	cn-hangzhou	区域ID。

返回数据

名称	类型	示例值	描述
RequestId	String	0B3BAFBB-3C0B-43B2-8C5E-3FA93173E160	请求ID。
RunSummary	Struct		运行概要信息。
ActualState	String	RUNNING	实际运行状态： • WAITING：等待。 • RUNNING：运行。 • PAUSED：暂停。 • TERMINATED：停止。 • SUCCESS：成功（批作业）。 • FAILED：失败（流作业）。
EngineJobHandler	String	application_1561366511517_113805|a10928cec91de395ff3d7593f7ff46c5	Instance在YARN中的名称： ApplicaitonID|JobID（JM分配的JobID）。

名称	类型	示例值	描述
ExpectState	String	RUNNING	期望运行状态： • WAITING：等待。 • RUNNING：运行。 • PAUSED：暂停。 • TERMINATED：停止。 • SUCCESS：成功（批作业）。 • FAILED：失败（流作业）。
Id	Long	214650	实例ID。
InputDelay	Long	0	业务延时。
JobName	String	job1	作业名称。
LastErrorMessage	String	error	最近错误信息。
LastErrorTime	Long	1548397575000	最近错误时间。

示例

请求示例

```
/api/v2/projects/project1/jobs/job1/instances/-1/runsummary
```

正常返回示例

`XML` 格式

```
<RequestId>0B3BAFBB-3C0B-43B2-8C5E-3FA93173E160</RequestId>
<RunSummary>
    <ActualState>RUNNING</ActualState>
    <EngineJobHandler>application_1561366511517_113805|a10928cec91de395ff3d7593f7ff46c5</EngineJobHandler>
    <ExpectState>RUNNING</ExpectState>
    <JobName>job1</JobName>
    <Id>214650</Id>
    <InputDelay>0</InputDelay>
</RunSummary>
```

`JSON` 格式

```
{
 "RequestId": "0B3BAFBB-3C0B-43B2-8C5E-3FA93173E160",
 "RunSummary": {
  "ActualState": "RUNNING",
  "EngineJobHandler": "application_1561366511517_113805|a10928cec91de395ff3d7593f7ff46c5",
  "ExpectState": "RUNNING",
  "JobName": "job1",
  "Id": 214650,
  "InputDelay": 0
 }
}
```

5.11. ListInstance

您可以通过ListInstance获取指定项目下所有的运行实例信息。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
GET /api/v2/projects/[projectName]/instances HTTP/1.1
```

请求参数

名称	类型	位置	是否必选	示例值	描述
jobType	String	Query	是	FLINK_STREAM	作业类型： • FLINK_STREAM（流作业）。 • FLINK_BATCH（批作业）。
projectName	String	Path	是	project1	项目名称。
RegionId	String	Header	否	cn-hangzhou	区域ID。
jobName	String	Query	否	job1	作业名称。

名称	类型	位置	是否必选	示例值	描述
startBeginTs	Long	Query	否	1548397575000	启动时间范围上限，和启动时间下限配合使用，搜索启动时间在该范围内的运行实例。使用13位时间戳，单位到毫秒。
startEndTs	Long	Query	否	1548397575000	启动时间范围下限，和启动时间上限配合使用，搜索启动时间在该范围内的运行实例。使用13位时间戳，单位到毫秒。
endBeginTs	Long	Query	否	1548397575000	停止时间范围上限，和停止时间下限配合使用，搜索停止时间在该范围内的运行实例。使用13位时间戳，单位到毫秒。
endEndTs	Long	Query	否	1548397575000	停止时间范围下限，和停止时间上限配合使用，搜索停止时间在该范围内的运行实例。使用13位时间戳，单位到毫秒。
apiType	String	Query	否	SQL	AP类型：DATASTREAM或SQL。
expectState	String	Query	否	RUNNING	期望状态： • RUNNING：运行。 • PAUSED：暂停。 • TERMINATED：停止。 • SUCCESS：成功（批作业）。 • FAILED：失败（流作业）。
actualState	String	Query	否	RUNNING	实际运行状态： • RUNNING：运行。 • PAUSED：暂停。 • TERMINATED：停止。 • SUCCESS：成功（批作业）。 • FAILED：失败（流作业）。

名称	类型	位置	是否必选	示例值	描述
isArchived	Boolean	Query	否	true	是否搜索归档实例，默认搜索非归档实例。对于流作业，同时只存在一个实例，当作业启动或者恢复运行的时候，将生成新的实例，旧实例会被归档；对于批作业，运行结束两天的实例会被归档。
pageSize	Integer	Query	否	10	分页选项，每页的实例数。
pageIndex	Integer	Query	否	1	分页选项，第几页。

返回数据

名称	类型	示例值	描述
Instances	Array of Instance		实例详情。
Instance			
ActualState	String	RUNNING	实际运行状态： • RUNNING：运行。 • PAUSED：暂停。 • TERMINATED：停止。 • SUCCESS：成功（批作业）。 • FAILED：失败（流作业）。
ApiType	String	SQL	API类型：DATASTREAM或SQL。
ClusterId	String	et2cloud****	集群ID。
Code	String	CREATE TABLE `test_output` (\n c1 BIGINT,\n c2 BIGINT,\n c3 VARCHAR\n) WITH (\n type = 'print'\n);\n\nINSERT INTO `test_output`\n select\n *\n from `test_input`;	实例代码。

名称	类型	示例值	描述
EndTime	Long	1548397575000	结束时间。
EngineJobHandler	String	application_1609851266735_7475|00bbc29106f9bf321048f85e5e57cd4c	Instance在YARN中的名称： ApplicaitonID | JobID（JM分配的JobID）。
EngineVersion	String	blink-3.5.0-hotfix	引擎版本。
ExpectState	String	RUNNING	期望状态： • RUNNING：运行。 • PAUSED：暂停。 • TERMINATED：停止。 • SUCCESS：成功（批作业）。 • FAILED：失败（流作业）。
Id	Long	589016	实例ID。
InputDelay	Long	0	业务延时。
JobName	String	job1	作业名称。
JobType	String	FLINK_STREAM	作业类型： • FLINK_STREAM（流作业）。 • FLINK_BATCH（批作业）。
LastErrorMessage	String	error	最近一次错误信息。
LastErrorTime	Long	1612336822000	最近错误时间。
LastOperateTime	Long	1612336822000	最近操作时间。
LastOperator	String	170906468757****	最近操作者。
Packages	String	package1.jar	Instance引用的Package，多个Package用逗号（,）分隔，未引用为空。

名称	类型	示例值	描述
PlanJson	String	{\n \"global\" : [{\n \"jobManagerMinCpuCores\" : 0.1,\n \"jobManagerMinMemoryCores\" : 1024,\n \"jobManagerCpuCores\" : 0.25,\n \"jobManagerMemoryInMB\" : 1024\n }],\n \"nodes\" : [{\n \"id\" : 1,\n \"uid\" : \"1\"	作业上线执行计划。
Priority	Integer	5	123
ProjectName	String	project1	项目名称。
Properties	String	#state.backend.rocksdb.ttl.ms=129600000\n\n#启用项目参数\n#disable.project.config=false\nblink.job.timeZone=Asia/Shanghai	作业运行配置参数。
QueueName	String	root.project1	队列名称。
StartTime	Long	1612336822000	作业启动时间。
Starter	String	170906468757****	启动人。
PageIndex	Integer	1	分页属性，第几页。
PageSize	Integer	10	分页属性，每页多少实例。
RequestId	String	33DBCFDA-5D75-452C-8DB3-5964CFE7747C	请求ID。
TotalCount	Long	15	总实例数目。
TotalPage	Integer	2	总页数。

示例

请求示例

```
/api/v2/projects/project1/instances
```

正常返回示例

`XML` 格式

```
<Instances>
    <Instance>
        <ExpectState>RUNNING</ExpectState>
        <EngineVersion>blink-3.5.0-hotfix</EngineVersion>
        <ProjectName>project1</ProjectName>
        <ClusterId>et2cloud****</ClusterId>
        <Priority>5</Priority>
        <PlanJson>{
  "global" : [ {
    "jobManagerMinCpuCores" : 0.1,
    "jobManagerMinMemoryCores" : 1024,
    "jobManagerCpuCores" : 0.25,
    "jobManagerMemoryInMB" : 1024
  } ],
  "nodes" : [ {
    "id" : 1,
    "uid" : "1",
    "name" : "RandomSource-test_input-Stream",
    "pact" : "Source",
    "chainingStrategy" : "HEAD",
    "parallelism" : 1,
    "maxParallelism" : 32768,
    "vcore" : 0.25,
    "heap_memory" : 252,
    "native_memory" : 4
  }, {
    "id" : 2,
    "uid" : "2",
    "name" : "SourceConversion(table:[builtin, default, test_input, source: [RandomSource-test_input]], fields:(s1, s2, s3))",
    "pact" : "Operator",
    "parallelism" : 1,
    "maxParallelism" : 32768,
    "vcore" : 0.25,
    "heap_memory" : 252,
    "native_memory" : 4
  }, {
    "id" : 3,
    "uid" : "3",
    "name" : "SinkConversion to Tuple2",
    "pact" : "Operator",
    "parallelism" : 1,
    "maxParallelism" : 32768,
    "vcore" : 0.25,
```

```
    "heap_memory" : 252,
    "native_memory" : 4
  }, {
    "id" : 4,
    "uid" : "4",
    "name" : "Flat Map",
    "pact" : "Operator",
    "parallelism" : 1,
    "maxParallelism" : 32768,
    "vcore" : 0.25,
    "heap_memory" : 252,
    "native_memory" : 4
  }, {
    "id" : 5,
    "uid" : "5",
    "name" : "Print to System.out",
    "pact" : "Sink",
    "parallelism" : 1,
    "maxParallelism" : 32768,
    "vcore" : 0.25,
    "heap_memory" : 252,
    "native_memory" : 4
  } ],
  "links" : [ {
    "source" : 1,
    "target" : 2
  }, {
    "source" : 2,
    "target" : 3
  }, {
    "source" : 3,
    "target" : 4
  }, {
    "source" : 4,
    "target" : 5
  } ]
}
</PlanJson>
        <JobName>job1</JobName>
        <StartTime>1612336822000</StartTime>
        <Properties>#checkpoint模式：EXACTLY_ONCE 或者 AT_LEAST_ONCE
#blink.checkpoint.mode=EXACTLY_ONCE
#checkpoint间隔时间，单位毫秒
#blink.checkpoint.interval.ms=180000
#rocksdb的数据生命周期，单位毫秒
#state.backend.rocksdb.ttl.ms=129600000
#启用项目参数
#disable.project.config=false
blink.job.timeZone=Asia/Shanghai</Properties>
        <Starter>170906468757****</Starter>
        <InputDelay>0</InputDelay>
        <Code>--SQL
--********************************************************************--
--Author: streamcompute@aliyun-test.com
--CreateTime: 2020-03-05 19:47:13
```

```
  CreateTime: 2020-05-05 19:47:15
--Comment: 请输入业务注释信息
--********************************************************************--
CREATE TABLE `test_input` (
  s1  BIGINT,
  s2  BIGINT,
  s3  VARCHAR
) WITH (
  `type` = 'random'
);
CREATE TABLE `test_output` (
  c1 BIGINT,
  c2 BIGINT,
  c3 VARCHAR
) WITH (
  type = 'print'
);
INSERT INTO `test_output`
  select
   *
  from `test_input`;</Code>
        <ActualState>RUNNING</ActualState>
        <EngineJobHandler>application_1609851266735_7475|00bbc29106f9bf321048f85e5e57cd4c</
EngineJobHandler>
        <JobType>FLINK_STREAM</JobType>
        <LastOperator>170906468757****</LastOperator>
        <Packages/>
        <ApiType>SQL</ApiType>
        <Id>589016</Id>
        <LastOperateTime>1612336822000</LastOperateTime>
        <QueueName>root.bayes_team</QueueName>
    </Instance>
    <Instance>
        <ExpectState>RUNNING</ExpectState>
        <EngineVersion>blink-2.2.6</EngineVersion>
        <ProjectName>bayes_team</ProjectName>
        <ClusterId>et2cloudprod</ClusterId>
        <Priority>5</Priority>
        <PlanJson>{
  "resourceSettings" : {
    "blink.resource.unit.per.worker" : 1,
    "blink.resource.unit.granularity" : 0.1,
    "blink.metrics.endpoint" : "http://100.118.128.166:9000/api/query?tenant=blink&amp;toke
n=e8ff43ab",
    "blink.resource.allocation.jm.min.cpu.cores" : 0.2,
    "blink.resource.allocation.jm.min.memory.mb" : 512
  },
  "autoConfig" : {
    "goal" : {
      "maxResourceUnits" : 10000.0
    },
    "result" : {
      "attemptId" : 1,
      "attempts" : 1,
      "attemptsCurrentPlan" : 1,
```

```
      "statusMessage" : "New job, using default configuration.",
      "scalingAction" : "InitialScale",
      "allocatedResourceUnits" : 0.9000000000000001,
      "allocatedCpuCores" : 0.9000000000000001,
      "allocatedMemoryInMB" : 3685
    }
  },
  "global" : [ {
    "tracingMetricsEnabled" : true,
    "jobManagerCpuCores" : 0.30000000000000004,
    "jobManagerMemoryInMB" : 1228
  } ],
  "nodes" : [ {
    "id" : 1,
    "uid" : "1",
    "pact" : "Data Source",
    "name" : "GSlsTableSource-sls_source-Stream",
    "chainingStrategy" : "HEAD",
    "parallelism" : 2,
    "maxParallelism" : 32768,
    "vcore" : 0.1,
    "heap_memory" : 128
  }, {
    "id" : 2,
    "uid" : "2",
    "pact" : "Operator",
    "name" : "SourceConversion(table:[_DataStreamTable_0, source: [GSlsTableSource-sls_sour
ce]], fields:(f0))",
    "parallelism" : 2,
    "maxParallelism" : 32768,
    "vcore" : 0.1,
    "heap_memory" : 128
  }, {
    "id" : 3,
    "uid" : "3",
    "pact" : "Operator",
    "name" : "correlate: table(SlsParser0($cor0.f0)), select: name",
    "parallelism" : 2,
    "maxParallelism" : 32768,
    "vcore" : 0.1,
    "heap_memory" : 128
  }, {
    "id" : 5,
    "uid" : "5",
    "pact" : "Operator",
    "name" : "GroupAggregate(groupBy: (name), select: (name, COUNT(name) AS EXPR$0))",
    "chainingStrategy" : "HEAD",
    "parallelism" : 2,
    "maxParallelism" : 32768,
    "vcore" : 0.1,
    "heap_memory" : 128
  }, {
    "id" : 6,
    "uid" : "6",
```

```
    "pact" : "Operator",
    "name" : "Calc(select: (EXPR$0, name))",
    "parallelism" : 2,
    "maxParallelism" : 32768,
    "vcore" : 0.1,
    "heap_memory" : 128
  }, {
    "id" : 7,
    "uid" : "7",
    "pact" : "Operator",
    "name" : "SinkConversion to Tuple2",
    "parallelism" : 2,
    "maxParallelism" : 32768,
    "vcore" : 0.1,
    "heap_memory" : 128
  }, {
    "id" : 8,
    "uid" : "8",
    "pact" : "Operator",
    "name" : "Flat Map",
    "parallelism" : 2,
    "maxParallelism" : 32768,
    "vcore" : 0.1,
    "heap_memory" : 128
  }, {
    "id" : 9,
    "uid" : "9",
    "pact" : "Data Sink",
    "name" : "OutputFormatAdapterSink:com.alibaba.blink.streaming.connector.sls.sink.SlsOut
putFormat@ccd341d",
    "parallelism" : 2,
    "maxParallelism" : 32768,
    "vcore" : 0.1,
    "heap_memory" : 128
  } ],
  "links" : [ {
    "source" : 1,
    "target" : 2,
    "side" : "second"
  }, {
    "source" : 2,
    "target" : 3,
    "side" : "second"
  }, {
    "source" : 3,
    "target" : 5,
    "ship_strategy" : "HASH",
    "side" : "second"
  }, {
    "source" : 5,
    "target" : 6,
    "side" : "second"
  }, {
    "source" : 6,
```

```
    "target" : 7,
    "side" : "second"
  }, {
    "source" : 7,
    "target" : 8,
    "side" : "second"
  }, {
    "source" : 8,
    "target" : 9,
    "side" : "second"
  } ],
  "statefulNodes" : [ 5 ],
  "vertexAdjustments" : {
    "0" : {
      "parallelismFactor" : 2.0,
      "parallelismLimit" : 2
    },
    "1" : {
      "parallelismFactor" : 2.0
    }
  },
  "vertices" : {
    "0" : {
      "vertexId" : 0,
      "source" : true,
      "name" : "Source: GSlsTableSource-sls_source-Stream -&gt; SourceConversion(table:[_Da
taStreamTable_0, source: [GSlsTableSource-sls_source]], fields:(f0)) -&gt; correlate: table
(SlsParser0($cor0.f0)), select: name",
      "parallelismFactor" : 2.0,
      "parallelismLimit" : 2,
      "parallelism" : 2,
      "maxParallelism" : 32768,
      "maxPartitionsPerSourceParallelism" : 8,
      "minResource" : {
        "cpuCores" : 0.1,
        "heapMemoryInMB" : 384,
        "managedMemoryInMB" : 0,
        "directMemoryInMB" : 0,
        "nativeMemoryInMB" : 0
      },
      "transformations" : [ 1, 2, 3 ],
      "inputVertexIds" : [ ]
    },
    "1" : {
      "vertexId" : 1,
      "name" : "GroupAggregate(groupBy: (name), select: (name, COUNT(name) AS EXPR$0)) -&gt
; Calc(select: (EXPR$0, name)) -&gt; SinkConversion to Tuple2 -&gt; Flat Map -&gt; Sink: Ou
tputFormatAdapterSink:com.alibaba.blink.streaming.connector.sls.sink.SlsOutputFormat@ccd341
d",
      "parallelismFactor" : 2.0,
      "parallelismLimit" : 32768,
      "parallelism" : 2,
      "maxParallelism" : 32768,
      "minResource" : {
```

```
        "cpuCores" : 0.1,
        "heapMemoryInMB" : 640,
        "managedMemoryInMB" : 0,
        "directMemoryInMB" : 0,
        "nativeMemoryInMB" : 0
      },
      "transformations" : [ 5, 6, 7, 8, 9 ],
      "inputVertexIds" : [ 0 ]
    }
  },
  "subDags" : {
    "0" : {
      "vertices" : [ 0, 1 ],
      "inputFromVertices" : [ ],
      "targetTps" : 0.0
    }
  },
  "workers" : [ {
    "numWorkers" : 1,
    "resourceUnits" : 0.2,
    "estimatedCpuCores" : 0.2,
    "estimatedMemoryInMB" : 768.0,
    "allocatedCpuCores" : 0.2,
    "allocatedMemoryInMB" : 819,
    "taskSlots" : [ {
      "topologyID" : 0,
      "count" : 2
    } ]
  }, {
    "numWorkers" : 1,
    "resourceUnits" : 0.4,
    "estimatedCpuCores" : 0.2,
    "estimatedMemoryInMB" : 1280.0,
    "allocatedCpuCores" : 0.4,
    "allocatedMemoryInMB" : 1638,
    "taskSlots" : [ {
      "topologyID" : 1,
      "count" : 2
    } ]
  } ]
}
</PlanJson>
        <JobName>ss2</JobName>
        <StartTime>1575353621000</StartTime>
        <Properties>#checkpoint模式：EXACTLY_ONCE 或者 AT_LEAST_ONCE
#blink.checkpoint.mode=EXACTLY_ONCE
#checkpoint间隔时间，单位毫秒
#blink.checkpoint.interval.ms=180000
#rocksdb的数据生命周期，单位毫秒
#state.backend.rocksdb.ttl.ms=129600000
blink.job.timeZone=Asia/Shanghai</Properties>
        <Starter>1709064687573327</Starter>
        <InputDelay>0</InputDelay>
        <Code>--Blink SQL
--********************************************************************--
```

```
--Author: streamcompute@aliyun-test.com
--CreateTime: 2018-12-25 14:49:51
--Comment: 请输入业务注释信息
--************************************************************************--
CREATE TABLE sls_source (
    `name` VARCHAR,
    num VARCHAR,
    id BIGINT
) WITH (
 type= 'sls',
 endPoint = 'http://cn-hangzhou-corp.sls.aliyuncs.com',
 roleArn='acs:ram::1709064687573327:role/aliyunstreamdefaultrole',
 project = 'bayes-test-jituan',
 logStore = 'sls_source'
);
CREATE TABLE sls_output (
 id              BIGINT,
 `name`          varchar
) WITH (
 type='SLS',
    endPoint= 'http://cn-hangzhou-corp.sls.aliyuncs.com',
    roleArn='acs:ram::1709064687573327:role/aliyunstreamdefaultrole',
    project='bayes-test-jituan',
    logStore='sls_output'
);
INSERT INTO sls_output
SELECT
   COUNT(`name`),
   `name`
FROM sls_source GROUP BY `name`</Code>
        <ActualState>RUNNING</ActualState>
        <EngineJobHandler>application_1561366511517_113805|a10928cec91de395ff3d7593f7ff46c5
</EngineJobHandler>
        <JobType>FLINK_STREAM</JobType>
        <LastOperator>1709064687573327</LastOperator>
        <Packages/>
        <ApiType>SQL</ApiType>
        <Id>214650</Id>
        <LastOperateTime>1575353621000</LastOperateTime>
        <QueueName>root.bayes_team</QueueName>
    </Instance>
</Instances>
<TotalCount>15</TotalCount>
<RequestId>33DBCFDA-5D75-452C-8DB3-5964CFE7747C</RequestId>
<PageSize>10</PageSize>
<TotalPage>2</TotalPage>
<PageIndex>1</PageIndex>
```

`JSON` 格式

```
{
 "Instances": {
  "Instance": [
```

```
    {
     "ExpectState": "RUNNING",
     "EngineVersion": "blink-3.5.0-hotfix",
     "ProjectName": "project1",
     "ClusterId": "et2cloud****",
     "Priority": 5,
     "PlanJson": "{\n  \"global\" : [ {\n    \"jobManagerMinCpuCores\" : 0.1,\n    \"jobMana
gerMinMemoryCores\" : 1024,\n    \"jobManagerCpuCores\" : 0.25,\n    \"jobManagerMemoryInMB
\" : 1024\n  } ],\n  \"nodes\" : [ {\n    \"id\" : 1,\n    \"uid\" : \"1\",\n    \"name\" :
\"RandomSource-test_input-Stream\",\n    \"pact\" : \"Source\",\n    \"chainingStrategy\" :
\"HEAD\",\n    \"parallelism\" : 1,\n    \"maxParallelism\" : 32768,\n    \"vcore\" : 0.25,
\n    \"heap_memory\" : 252,\n    \"native_memory\" : 4\n  }, {\n    \"id\" : 2,\n    \"uid
\" : \"2\",\n    \"name\" : \"SourceConversion(table:[builtin, default, test_input, source:
[RandomSource-test_input]], fields:(s1, s2, s3))\",\n    \"pact\" : \"Operator\",\n    \"pa
rallelism\" : 1,\n    \"maxParallelism\" : 32768,\n    \"vcore\" : 0.25,\n    \"heap_memory
\" : 252,\n    \"native_memory\" : 4\n  }, {\n    \"id\" : 3,\n    \"uid\" : \"3\",\n    \"
name\" : \"SinkConversion to Tuple2\",\n    \"pact\" : \"Operator\",\n    \"parallelism\" :
1,\n    \"maxParallelism\" : 32768,\n    \"vcore\" : 0.25,\n    \"heap_memory\" : 252,\n
\"native_memory\" : 4\n  }, {\n    \"id\" : 4,\n    \"uid\" : \"4\",\n    \"name\" : \"Flat
Map\",\n    \"pact\" : \"Operator\",\n    \"parallelism\" : 1,\n    \"maxParallelism\" : 32
768,\n    \"vcore\" : 0.25,\n    \"heap_memory\" : 252,\n    \"native_memory\" : 4\n  }, {\
n    \"id\" : 5,\n    \"uid\" : \"5\",\n    \"name\" : \"Print to System.out\",\n    \"pact
\" : \"Sink\",\n    \"parallelism\" : 1,\n    \"maxParallelism\" : 32768,\n    \"vcore\" :
0.25,\n    \"heap_memory\" : 252,\n    \"native_memory\" : 4\n  } ],\n  \"links\" : [ {\n
\"source\" : 1,\n    \"target\" : 2\n  }, {\n    \"source\" : 2,\n    \"target\" : 3\n  },
{\n    \"source\" : 3,\n    \"target\" : 4\n  }, {\n    \"source\" : 4,\n    \"target\" : 5
\n  } ]\n}\n",
     "JobName": "job1",
     "StartTime": 1612336822000,
     "Properties": "#checkpoint模式：EXACTLY_ONCE 或者 AT_LEAST_ONCE\n#blink.checkpoint.mode=
EXACTLY_ONCE\n\n#checkpoint间隔时间，单位毫秒\n#blink.checkpoint.interval.ms=180000\n\n#rocks
db的数据生命周期，单位毫秒\n#state.backend.rocksdb.ttl.ms=129600000\n\n#启用项目参数\n#disable.p
roject.config=false\nblink.job.timeZone=Asia/Shanghai",
     "Starter": "170906468757****",
     "InputDelay": 0,
     "Code": "--SQL\n--************************************************************************-
-\n--Author: streamcompute@aliyun-test.com\n--CreateTime: 2020-03-05 19:47:13\n--Comment:
请输入业务注释信息\n--************************************************************************--\
n\nCREATE TABLE `test_input` (\n  s1  BIGINT,\n  s2  BIGINT,\n  s3  VARCHAR\n) WITH (\n  `t
ype` = 'random'\n);\n\nCREATE TABLE `test_output` (\n  c1 BIGINT,\n  c2 BIGINT,\n  c3 VARCH
AR\n) WITH (\n  type = 'print'\n);\n\nINSERT INTO `test_output`\n  select\n  *\n  from `te
st_input`;",
     "ActualState": "RUNNING",
     "EngineJobHandler": "application_1609851266735_7475|00bbc29106f9bf321048f85e5e57cd4c",
     "JobType": "FLINK_STREAM",
     "LastOperator": "170906468757****",
     "Packages": "",
     "ApiType": "SQL",
     "Id": 589016,
     "LastOperateTime": 1612336822000,
     "QueueName": "root.bayes_team"
    },
    {
     "ExpectState": "RUNNING",
```

```
    "EngineVersion": "blink-2.2.6",
    "ProjectName": "bayes_team",
    "ClusterId": "et2cloudprod",
    "Priority": 5,
    "PlanJson": "{\n  \"resourceSettings\" : {\n    \"blink.resource.unit.per.worker\" : 1,
\n    \"blink.resource.unit.granularity\" : 0.1,\n    \"blink.metrics.endpoint\" : \"http:/
/100.118.128.166:9000/api/query?tenant=blink&token=e8ff43ab\",\n    \"blink.resource.alloca
tion.jm.min.cpu.cores\" : 0.2,\n    \"blink.resource.allocation.jm.min.memory.mb\" : 512\n
},\n  \"autoConfig\" : {\n    \"goal\" : {\n      \"maxResourceUnits\" : 10000.0\n    },\n
\"result\" : {\n      \"attemptId\" : 1,\n      \"attempts\" : 1,\n      \"attemptsCurrentP
lan\" : 1,\n      \"statusMessage\" : \"New job, using default configuration.\",\n      \"s
calingAction\" : \"InitialScale\",\n      \"allocatedResourceUnits\" : 0.9000000000000001,\
n      \"allocatedCpuCores\" : 0.9000000000000001,\n      \"allocatedMemoryInMB\" : 3685\n
}\n  },\n  \"global\" : [ {\n    \"tracingMetricsEnabled\" : true,\n    \"jobManagerCpuCore
s\" : 0.30000000000000004,\n    \"jobManagerMemoryInMB\" : 1228\n  } ],\n  \"nodes\" : [ {\
n    \"id\" : 1,\n    \"uid\" : \"1\",\n    \"pact\" : \"Data Source\",\n    \"name\" : \"G
SlsTableSource-sls_source-Stream\",\n    \"chainingStrategy\" : \"HEAD\",\n    \"parallelis
m\" : 2,\n    \"maxParallelism\" : 32768,\n    \"vcore\" : 0.1,\n    \"heap_memory\" : 128\
n  }, {\n    \"id\" : 2,\n    \"uid\" : \"2\",\n    \"pact\" : \"Operator\",\n    \"name\"
: \"SourceConversion(table:[_DataStreamTable_0, source: [GSlsTableSource-sls_source]], fiel
ds:(f0))\",\n    \"parallelism\" : 2,\n    \"maxParallelism\" : 32768,\n    \"vcore\" : 0.1
,\n    \"heap_memory\" : 128\n  }, {\n    \"id\" : 3,\n    \"uid\" : \"3\",\n    \"pact\" :
\"Operator\",\n    \"name\" : \"correlate: table(SlsParser0($cor0.f0)), select: name\",\n
\"parallelism\" : 2,\n    \"maxParallelism\" : 32768,\n    \"vcore\" : 0.1,\n    \"heap_mem
ory\" : 128\n  }, {\n    \"id\" : 5,\n    \"uid\" : \"5\",\n    \"pact\" : \"Operator\",\n
\"name\" : \"GroupAggregate(groupBy: (name), select: (name, COUNT(name) AS EXPR$0))\",\n
\"chainingStrategy\" : \"HEAD\",\n    \"parallelism\" : 2,\n    \"maxParallelism\" : 32768,
\n    \"vcore\" : 0.1,\n    \"heap_memory\" : 128\n  }, {\n    \"id\" : 6,\n    \"uid\" : \
"6\",\n    \"pact\" : \"Operator\",\n    \"name\" : \"Calc(select: (EXPR$0, name))\",\n
\"parallelism\" : 2,\n    \"maxParallelism\" : 32768,\n    \"vcore\" : 0.1,\n    \"heap_mem
ory\" : 128\n  }, {\n    \"id\" : 7,\n    \"uid\" : \"7\",\n    \"pact\" : \"Operator\",\n
\"name\" : \"SinkConversion to Tuple2\",\n    \"parallelism\" : 2,\n    \"maxParallelism\"
: 32768,\n    \"vcore\" : 0.1,\n    \"heap_memory\" : 128\n  }, {\n    \"id\" : 8,\n    \"u
id\" : \"8\",\n    \"pact\" : \"Operator\",\n    \"name\" : \"Flat Map\",\n    \"parallelis
m\" : 2,\n    \"maxParallelism\" : 32768,\n    \"vcore\" : 0.1,\n    \"heap_memory\" : 128\
n  }, {\n    \"id\" : 9,\n    \"uid\" : \"9\",\n    \"pact\" : \"Data Sink\",\n    \"name\"
: \"OutputFormatAdapterSink:com.alibaba.blink.streaming.connector.sls.sink.SlsOutputFormat@
ccd341d\",\n    \"parallelism\" : 2,\n    \"maxParallelism\" : 32768,\n    \"vcore\" : 0.1,
\n    \"heap_memory\" : 128\n  } ],\n  \"links\" : [ {\n    \"source\" : 1,\n    \"target\"
: 2,\n    \"side\" : \"second\"\n  }, {\n    \"source\" : 2,\n    \"target\" : 3,\n    \"si
de\" : \"second\"\n  }, {\n    \"source\" : 3,\n    \"target\" : 5,\n    \"ship_strategy\"
: \"HASH\",\n    \"side\" : \"second\"\n  }, {\n    \"source\" : 5,\n    \"target\" : 6,\n
\"side\" : \"second\"\n  }, {\n    \"source\" : 6,\n    \"target\" : 7,\n    \"side\" : \"s
econd\"\n  }, {\n    \"source\" : 7,\n    \"target\" : 8,\n    \"side\" : \"second\"\n  },
{\n    \"source\" : 8,\n    \"target\" : 9,\n    \"side\" : \"second\"\n  } ],\n  \"statefu
lNodes\" : [ 5 ],\n  \"vertexAdjustments\" : {\n    \"0\" : {\n      \"parallelismFactor\"
: 2.0,\n      \"parallelismLimit\" : 2\n    },\n    \"1\" : {\n      \"parallelismFactor\"
: 2.0\n    }\n  },\n  \"vertices\" : {\n    \"0\" : {\n      \"vertexId\" : 0,\n      \"sou
rce\" : true,\n      \"name\" : \"Source: GSlsTableSource-sls_source-Stream -> SourceConver
sion(table:[_DataStreamTable_0, source: [GSlsTableSource-sls_source]], fields:(f0)) -> corr
elate: table(SlsParser0($cor0.f0)), select: name\",\n      \"parallelismFactor\" : 2.0,\n
\"parallelismLimit\" : 2,\n      \"parallelism\" : 2,\n      \"maxParallelism\" : 32768,\n
\"maxPartitionsPerSourceParallelism\" : 8,\n      \"minResource\" : {\n        \"cpuCores\"
: 0.1,\n        \"heapMemoryInMB\" : 384,\n        \"managedMemoryInMB\" : 0,\n        \"di
rectMemoryInMB\" : 0,\n        \"nativeMemoryInMB\" : 0\n      },\n      \"transformations\
```

```
" : [ 1, 2, 3 ],\n      \"inputVertexIds\" : [ ]\n    },\n    \"1\" : {\n      \"vertexId\"
: 1,\n      \"name\" : \"GroupAggregate(groupBy: (name), select: (name, COUNT(name) AS EXPR
$0)) -> Calc(select: (EXPR$0, name)) -> SinkConversion to Tuple2 -> Flat Map -> Sink: Outpu
tFormatAdapterSink:com.alibaba.blink.streaming.connector.sls.sink.SlsOutputFormat@ccd341d\"
,\n      \"parallelismFactor\" : 2.0,\n      \"parallelismLimit\" : 32768,\n      \"paralle
lism\" : 2,\n      \"maxParallelism\" : 32768,\n      \"minResource\" : {\n        \"cpuCor
es\" : 0.1,\n        \"heapMemoryInMB\" : 640,\n        \"managedMemoryInMB\" : 0,\n
\"directMemoryInMB\" : 0,\n        \"nativeMemoryInMB\" : 0\n      },\n      \"transformati
ons\" : [ 5, 6, 7, 8, 9 ],\n      \"inputVertexIds\" : [ 0 ]\n    }\n  },\n  \"subDags\" :
{\n    \"0\" : {\n      \"vertices\" : [ 0, 1 ],\n      \"inputFromVertices\" : [ ],\n
\"targetTps\" : 0.0\n    }\n  },\n  \"workers\" : [ {\n    \"numWorkers\" : 1,\n    \"resou
rceUnits\" : 0.2,\n    \"estimatedCpuCores\" : 0.2,\n    \"estimatedMemoryInMB\" : 768.0,\n
\"allocatedCpuCores\" : 0.2,\n    \"allocatedMemoryInMB\" : 819,\n    \"taskSlots\" : [ {\n
\"topologyID\" : 0,\n      \"count\" : 2\n    } ]\n  }, {\n    \"numWorkers\" : 1,\n    \"r
esourceUnits\" : 0.4,\n    \"estimatedCpuCores\" : 0.2,\n    \"estimatedMemoryInMB\" : 1280
.0,\n    \"allocatedCpuCores\" : 0.4,\n    \"allocatedMemoryInMB\" : 1638,\n    \"taskSlots
\" : [ {\n      \"topologyID\" : 1,\n      \"count\" : 2\n    } ]\n  } ]\n}\n",
    "JobName": "ss2",
    "StartTime": 1575353621000,
    "Properties": "#checkpoint模式：EXACTLY_ONCE 或者 AT_LEAST_ONCE\n#blink.checkpoint.mode=
EXACTLY_ONCE\n\n#checkpoint间隔时间，单位毫秒\n#blink.checkpoint.interval.ms=180000\n\n#rocks
db的数据生命周期，单位毫秒\n#state.backend.rocksdb.ttl.ms=129600000\nblink.job.timeZone=Asia/S
hanghai",
    "Starter": "1709064687573327",
    "InputDelay": 0,
    "Code": "--Blink SQL\n--********************************************************************
******--\n--Author: streamcompute@aliyun-test.com\n--CreateTime: 2018-12-25 14:49:51\n--Comm
ent: 请输入业务注释信息\n--********************************************************************
**--\n\nCREATE TABLE sls_source (\n    `name` VARCHAR,\n    num VARCHAR,\n    id BIGINT\n)
WITH (\n\ttype= 'sls',\n\tendPoint = 'http://cn-hangzhou-corp.sls.aliyuncs.com',\n\troleArn
='acs:ram::1709064687573327:role/aliyunstreamdefaultrole',\n\tproject = 'bayes-test-jituan'
,\n\tlogStore = 'sls_source'\n);\n\nCREATE TABLE sls_output (\n\tid              BIGINT,\n\
t`name`          varchar\n) WITH (\n\ttype='SLS',\n    endPoint= 'http://cn-hangzhou-corp.s
ls.aliyuncs.com',\n    roleArn='acs:ram::1709064687573327:role/aliyunstreamdefaultrole',\n
project='bayes-test-jituan',\n    logStore='sls_output'\n);\n\nINSERT INTO sls_output\nSELE
CT \n   COUNT(`name`),\n   `name`\nFROM sls_source GROUP BY `name`",
    "ActualState": "RUNNING",
    "EngineJobHandler": "application_1561366511517_113805|a10928cec91de395ff3d7593f7ff46c5"
,
    "JobType": "FLINK_STREAM",
    "LastOperator": "1709064687573327",
    "Packages": "",
    "ApiType": "SQL",
    "Id": 214650,
    "LastOperateTime": 1575353621000,
    "QueueName": "root.bayes_team"
   }
  ]
 },
 "TotalCount": 15,
 "RequestId": "33DBCFDA-5D75-452C-8DB3-5964CFE7747C",
 "PageSize": 10,
 "TotalPage": 2,
 "PageIndex": 1
```

```
}
```

5.12. ModifyInstanceState

您可以通过ModifyInstanceState修改运行实例的状态，实现对实例的操作。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
PUT /api/v2/projects/[projectName]/jobs/[jobName]/instances/[instanceId]/expectstate HTTPS
```

请求参数

名称	类型	是否必选	示例值	描述
expectState	String	是	RUNNING	期望状态： • RUNNING：运行中。 • PAUSED：暂停。 • TERMINATED：停止。 • SUCCESS：成功（批作业）。 • FAILED：失败（流作业）。
instanceId	Long	是	-1	实例ID： • 流作业：填写实际运行的实例ID或直接填写-1（当前运行的实例）。 • 批作业：可以通过ListInstance、StartJob等接口获得。
jobName	String	是	job1	作业名称
projectName	String	是	project1	项目名称
RegionId	String	否	cn-hangzhou	区域ID **说明** 公共云用户请忽略此参数。

名称	类型	是否必选	示例值	描述
isFlush	Boolean	否	true	在恢复作业时，是否使用最新配置。默认值为true。
triggerCheckpoint	Boolean	否	true	在暂停或者停止作业时，是否进行一次Checkpoint。 **说明** Blink 3.5.0及以上版本支持该参数。

返回数据

名称	类型	示例值	描述
RequestId	String	EA0059FA-FD69-4A83-B216-6B36B03129A7	请求ID

示例

请求示例

```
/api/v2/projects/project1/jobs/job1/instances/-1/expectstate
{\"expectState\":\"RUNNING\"}
```

正常返回示例

`XML` 格式

```
<requestId>EA0059FA-FD69-4A83-B216-6B36B03129A7</requestId>
```

`JSON` 格式

```
{
    "requestId": "EA0059FA-FD69-4A83-B216-6B36B03129A7"
}
```

5.13. GetInstanceHistoryAutoScalePlanContent

您可以通过GetInstanceHistoryAutoScalePlanContent查看到历史调优某次PlanJson的详细内容，需要和获取作业历史调优PlanJson列表配合使用。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
GET /api/v2/projects/[projectName]/jobs/[jobName]/instance/[instanceId]/autoscale/plancontent HTTP/1.1
```

请求参数

名称	类型	位置	是否必选	示例值	描述
instanceId	Long	Path	是	-1	作业实例ID。流作业填写-1，批作业不支持使用。
jobName	String	Path	是	job1	作业名称。
planName	String	Query	是	2018-09-18 10:10:20	自动调优某次PlanJson的名称。
projectName	String	Path	是	project1	项目名称。
RegionId	String	Header	是	cn-shanghai	区域ID。

返回数据

名称	类型	示例值	描述
PlanContent	String	{k:v}	PlanJson的详细内容。
RequestId	String	6C799EEF-1DD8-4C55-961C-3B9D55A87AE6	请求ID。

示例

请求示例

```
/api/v2/projects/[projectName]/jobs/[jobName]/instance/[instanceId]/autoscale/plancontent
```

正常返回示例

`XML` 格式

```
<RequestId>6C799EEF-1DD8-4C55-961C-3B9D55A87AE6</RequestId>
<Message>Specified parameter instanceId is not valid.</Message>
<Recommend>https://error-center.aliyun.com/status/search?Keyword=InvalidinstanceId&amp;sour
ce=PopGw</Recommend>
<HostId>foas.cn-shanghai.aliyuncs.com</HostId>
<Code>InvalidinstanceId</Code>
```

`JSON` 格式

```
{
 "RequestId": "6C799EEF-1DD8-4C55-961C-3B9D55A87AE6",
 "Message": "Specified parameter instanceId is not valid.",
 "Recommend": "https://error-center.aliyun.com/status/search?Keyword=InvalidinstanceId&sour
ce=PopGw",
 "HostId": "foas.cn-shanghai.aliyuncs.com",
 "Code": "InvalidinstanceId"
}
```

5.14. GetInstanceHistoryAutoScalePlanList

您可以通过GetInstanceHistoryAutoScalePlanList获取作业自动调优PlanJson列表。配合GetInstanceHistoryAutoScalePlanContent接口可以定位到指定的PlanJson。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
GET /api/v2/projects/[projectName]/jobs/[jobName]/instance/[instanceId]/autoscale/planlist
HTTP/1.1
```

请求参数

名称	类型	位置	是否必选	示例值	描述
instanceId	Long	Path	是	-1	作业实例ID。流作业参数值是 `-1`，批作业暂不支持。
jobName	String	Path	是	job1	作业名称。
projectName	String	Path	是	project1	项目名称。

名称	类型	位置	是否必选	示例值	描述
RegionId	String	Header	是	cn-shanghai	区域ID

返回数据

名称	类型	示例值	描述
PlanList	List	[planA,planB]	自动调优作业历史Plan名称。
RequestId	String	9C010057-9F22-45DC-A415-4E4E53181E5E	请求ID。

示例

请求示例

```
/api/v2/projects/project1/jobs/job1/instance/-1/autoscale/planlist
```

正常返回示例

`XML` 格式

```
<RequestId>9C010057-9F22-45DC-A415-4E4E53181E5E</RequestId>
```

`JSON` 格式

```
{
 "RequestId": "9C010057-9F22-45DC-A415-4E4E53181E5E"
}
```

6.文件夹

6.1. CreateFolder

您可以通过CreateFolder OpenAPI在项目对应路径下创建文件夹，并在开发控制台上保存文件夹ID和作业路径。

> **说明** 作业开发对应根目录，在开发控制台上，作业都显示在作业开发目录下。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
POST /api/v2/projects/[projectName]/folders HTTPS
```

请求参数

名称	类型	是否必选	示例值	描述
path	String	是	/path	文件夹路径
projectName	String	是	project1	项目名称
RegionId	String	是	cn-hangzhou	区域ID **说明** 公共云用户请忽略此参数。

返回数据

名称	类型	示例值	描述
FolderId	Long	33193	文件夹ID

名称	类型	示例值	描述
RequestId	String	B43F695A-F126-46C0-B336-846D6290474D	请求ID

示例

请求示例

```
/api/v2/projects/project1/folders
"{\"path\":\"/path\"}"
```

正常返回示例

`XML` 格式

```
<RequestId>B43F695A-F126-46C0-B336-846D6290474D</RequestId>
<FolderId>33193</FolderId>
```

`JSON` 格式

```
{
 "RequestId": "B43F695A-F126-46C0-B336-846D6290474D",
 "FolderId": 33193
}
```

6.2. DeleteFolder

您可以通过DeleteFolder OpenAPI删除空文件夹，删除非空文件夹会报错。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
DELETE /api/v2/projects/[projectName]/folders HTTPS
```

请求参数

名称	类型	是否必选	示例值	描述
path	String	是	/path1/path2	文件夹路径

名称	类型	是否必选	示例值	描述
projectName	String	是	project1	项目名称
RegionId	String	否	cn-hangzhou	区域ID **说明** 公共云用户请忽略此参数。

返回数据

名称	类型	示例值	描述
RequestId	String	C5C6746D-2FFC-4D5D-A6A8-FA8DF7585DC2	请求ID

示例

请求示例

```
/api/v2/projects/project1/folders
```

正常返回示例

`XML` 格式

```
<requestId>C5C6746D-2FFC-4D5D-A6A8-FA8DF7585DC2</requestId>
```

`JSON` 格式

```
{"requestId":"C5C6746D-2FFC-4D5D-A6A8-FA8DF7585DC2"}
```

6.3. GetFolder

您可以通过GetFolder OpenAPI获取文件夹信息，即文件夹ID和文件夹路径。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
GET /api/v2/projects/[projectName]/folders HTTPS
```

请求参数

名称	类型	是否必选	示例值	描述
path	String	是	/path1	文件夹路径
projectName	String	是	project1	项目名称
RegionId	String	是	cn-hangzhou	区域ID **说明** 公共云用户请忽略此参数。

返回数据

名称	类型	示例值	描述
Folder	Struct		文件夹详情
FolderId	Long	27552	文件夹ID
Path	String	/path1	文件夹路径
RequestId	String	AFD7EF32-88FA-4539-A83C-EF91DDFA2C96	请求ID

示例

请求示例

```
/api/v2/projects/project1/folders
```

正常返回示例

`XML` 格式

```
<RequestId>AFD7EF32-88FA-4539-A83C-EF91DDFA2C96</RequestId>
<Folder>
    <Path>/path1</Path>
    <FolderId>27552</FolderId>
</Folder>
```

JSON 格式

```
{
 "RequestId": "AFD7EF32-88FA-4539-A83C-EF91DDFA2C96",
 "Folder": {
  "Path": "/path1",
  "FolderId": 27552
 }
}
```

6.4. ListChildFolder

您可以通过List ChildFolder OpenAPI获取指定主目录下所有子目录的ID和路径。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
GET /api/v2/projects/[projectName]/folders/children HTTPS
```

请求参数

名称	类型	是否必选	示例值	描述
path	String	是	/path1	主目录路径
projectName	String	是	project1	项目名称
RegionId	String	否	cn-hangzhou	区域ID **说明** 公共云用户请忽略此参数。

返回数据

名称	类型	示例值	描述
Folders	Array of Folder		文件目录详情

名称	类型	示例值	描述
Folder			
FolderId	Long	33215	文件夹ID
Path	String	/path1/path2	文件夹目录
RequestId	String	DA0A418B-E495-4BCC-B44F-1BA3CE731103	请求ID

示例

请求示例

```
/api/v2/projects/project1/folders/children
```

正常返回示例

`XML` 格式

```
<RequestId>DA0A418B-E495-4BCC-B44F-1BA3CE731103</RequestId>
<Folders>
    <Folder>
        <Path>/path1/path2</Path>
        <FolderId>33215</FolderId>
    </Folder>
</Folders>
```

`JSON` 格式

```
{
 "RequestId": "DA0A418B-E495-4BCC-B44F-1BA3CE731103",
 "Folders": {
  "Folder": [
   {
    "Path": "/path1/path2",
    "FolderId": 33215
   }
  ]
 }
}
```

6.5. MVFolder

您可以通过MVFolder OpenAPI移动指定项目下的文件夹。移动非空文件夹时，会将文件夹下的作业一起移动。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
PUT /api/v2/projects/[projectName]/folders HTTPS
```

请求参数

名称	类型	是否必选	示例值	描述
destPath	String	是	/path2	目标路径
projectName	String	是	project1	项目名称
srcPath	String	是	/path1	原路径
RegionId	String	否	cn-hangzhou	区域ID **说明** 公共云用户请忽略此参数。

返回数据

名称	类型	示例值	描述
RequestId	String	D34CBA74-DED8-4B1E-80B4-DE4C58291726	请求ID

示例

请求示例

```
/api/v2/projects/project1/folders
{\"srcPath\":\"/path1\",\"destPath\":\"/path2\"}
```

正常返回示例

`XML` 格式

```
<RequestId>D34CBA74-DED8-4B1E-80B4-DE4C58291726</RequestId>
```

`JSON` 格式

```
{
 "RequestId": "D34CBA74-DED8-4B1E-80B4-DE4C58291726"
}
```

7.作业

7.1. CheckRawPlanJson

您可以组合调用CheckRawPlanJson和GetRawPlanJson接口，来获取某个作业的PlanJson。CheckRawPlanJson接口发出调用请求后会返回一个Session ID，GetRawPlanJson获取该session ID后可以查询到PlanJson的详情。

> **说明** 建议循环调用CheckRawPlanJson直到成功获取。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
GET /api/v2/projects/[projectName]/jobs/[jobName]/planjson/check HTTP/1.1
```

请求参数

名称	类型	位置	是否必选	示例值	描述
jobName	String	Path	是	job1	作业名称。
projectName	String	Path	是	project1	项目名称。
RegionId	String	Header	是	cn-hangzhou	区域ID。
sessionId	String	Query	是	j6b43mm10nnzywsixmeh8maflgt6xq5afyeuflow3fravqlby0udi605mi7sty2pem2w9nysqiztag5je4esvmgq04pdcazgdthy3u5riy6na0fz6fmgph1k6b62bvjs7kqorn69ujsk0	当使用CheckRawPlanJson发送请求后，会返回一个SessionId，将该sessionId填在此处。

返回数据

名称	类型	示例值	描述
PlanJsonInfo	Struct		PlanJson详情。
ErrorMessage	String	eroor	错误信息。
PlanJson	String	{a:b}	执行计划。
Status	String	success	GetRawPlanJson的执行状态： • Sessionin run ： 执行中。 • success：成功。 • fail：失败。
RequestId	String	FD0FF8C0-779A-45EB-9674-FF3E127B10D2	请求ID。

示例

请求示例

```
/api/v2/projects/project1/jobs/project1/planjson/check
```

正常返回示例

`XML` 格式

```
<RequestId>78261AC6-B5C9-4514-B82F-39BCC45A5D63</RequestId>
```

`JSON` 格式

```
{
    "RequestId": "78261AC6-B5C9-4514-B82F-39BCC45A5D63"
}
```

7.2. CommitJob

您可以通过CommitJob提交作业。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
PUT /api/v2/projects/[projectName]/jobs/[jobName]/commit HTTPS
```

请求参数

名称	类型	位置	是否必选	示例值	描述
jobName	String	Path	是	job1	作业名称
projectName	String	Path	是	project1	项目名称
RegionId	String	Header	是	cn-hangzhou	区域ID **说明** 公共云用户请忽略此参数。
isOnOff	Boolean	Body	否	true	是否启用AutoScale。
maxCU	Float	Body	否	10.0	最大CU数，作业运行期间使用的资源不会超过该上限。单位为CU，`1 CU=1 Core + 4 GB RAM`。

名称	类型	位置	是否必选	示例值	描述
configure	String	Body	否	{"fetch_delay":1.0,"recommendOnly":true}	您可以配置以下几项： • 调优策略：系统根据选择的策略和期望值对作业自动调优，以达到期望值。目前只支持数据滞留时间fetch_delay，单位为秒。 • 期望值：数据滞留时间阈值，当Source的数据滞留时间超过该阈值时，触发AutoScale并优化作业并发度。 • 建议recommendOnly：true表示开启PlanJson建议功能。 **说明** • 仅支持通过AutoScale后台服务提供PlanJson建议。 • 仅在AutoScale设置为true时，recommendOnly生效。 • 仅Blink 3.7.8以上版本支持recommendOnly。 • 仅Blink 3.x版本支持AutoScale功能。 • 当开启AutoScale时，需要传入maxCu和configure参数。
recommendOnly	Boolean	Body	否	false	启用AutoScale后，系统是否自动修改您的PlanJson： • true：系统不会自动修改。 • false（默认值）：系统会自动修改。 **说明** 可以通过SDK中getInstanceHistoryAutoScalePlanListRequest获取PlanJson列表。
suspendPeriodParam.N.plan	String	Body	否	ORIGINAL	指定暂停作业后使用的PlanJson： • ORIGINAL：初始的PlanJson。 • CURRENT：当前的PlanJson。

名称	类型	位置	是否必选	示例值	描述
suspendPeriodParam.N.policy	String	Body	否	EVERY_DAY_TIME	指定暂停时间段： • EVERY_DAY_TIME：指定每天的暂停时间段，需要填写暂停起始时间（startTime:HH:mm）和暂停结束时间（endTime:HH:mm）。 • SPECIFIED_TIME：指定具体的暂停时间段，需要填写暂停起始时间（startTime:yyyy-MM-dd HH:mm）和暂停结束时间（endTime:yyyy-MM-dd HH:mm）。
suspendPeriodParam.N.startTime	String	Body	否	18:20	暂停时间段的开始时间。 **说明** 根据policy设置不同格式的时间格式。
suspendPeriodParam.N.endTime	String	Body	否	18:25	暂停时间段的结束时间。 **说明** 根据policy设置不同格式的时间格式。
suspendPeriods	String	Body	否	[{"endTime":"18:20","startTime":"18:00","plan":"ORIGINAL","policy":"EVERY_DAY_TIME"},{"endTime":"19:00","startTime":"18:40","plan":"CURRENT","policy":"EVERY_DAY_TIME"},{"endTime":"2020-07-17 20:00","startTime":"2020-07-17 19:30","plan":"CURRENT","policy":"SPECIFIED_TIME"}]	开启AutoScale后，指定暂停AutoScale时间段和使用的PlanJson。

返回数据

名称	类型	示例值	描述
RequestId	String	C5C6746D-2FFC-4D5D-A6A8-FA8DF7585DC2	请求ID

示例

请求示例

```
/api/v2/projects/project1/jobs/job1/commit
```

正常返回示例

`XML` 格式

```
<requestId>C5C6746D-2FFC-4D5D-A6A8-FA8DF7585DC2</requestId>
```

`JSON` 格式

```
{
    "requestId": "C5C6746D-2FFC-4D5D-A6A8-FA8DF7585DC2"
}
```

7.3. CreateJob

您可以通过CreateJob OpenAPI在指定项目下创建作业。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
POST /api/v2/projects/[projectName]/jobs HTTPS
```

请求参数

名称	类型	位置	是否必选	示例值	描述

名称	类型	位置	是否必选	示例值	描述
apiType	String	Body	是	Datastream	API类型： • DATASTREAM：Datastream作业 • SQL：SQL作业
clusterId	String	Body	是	rcmp9x37ztfb63g1x7lt****	集群ID **说明** 您可以使用listcluster获取集群ID信息。
code	String	Body	是	blink.main.class=com.hjc.test.blink_test.ChinanTopSpeedWindowing2\r\nblink.job.name=datastream_preste st	作业代码： • SQL作业：填写SQL代码。 • DataStream作业：填写DataStream作业的相关参数。
engineVersion	String	Body	是	blink_2.2.4	引擎版本
jobName	String	Body	是	job1	作业名称
jobType	String	Body	是	FLINK_STREAM	作业类型： • FLINK_STREAM：流作业 • FLINK_BATCH：批作业
planJson	String	Body	是	{a:b}	执行计划
projectName	String	Path	是	project1	项目名称
queueName	String	Body	是	root.default	队列名称
RegionId	String	Header	是	区域ID	区域ID **说明** 公共云用户请忽略此参数。
properties	String	Body	否	{k:v}	作业运行的相关参数

名称	类型	位置	是否必选	示例值	描述
packages	String	Body	否	package1,package2	JAR包名称，多个使用逗号（,）分隔。
description	String	Body	否	test	作业备注
folderId	Long	Body	否	123	文件夹ID

返回数据

名称	类型	示例值	描述
RequestId	String	B661F03F-0F9D-4EFF-A40F-4ED1519A549A	请求ID

示例

请求示例

```
/api/v2/projects/project1/jobs
{\"jobName\":\"job1\",\"jobType\":\"FLINK_STREAM\",\"apiType\":\"Datastream\",\"code\":\"blink.main.class=com.hjc.test.blink_test.ChinanTopSpeedWindowing2\r\nblink.job.name=datastream_prestest\",\"planJson\":\"{a:b}\",\"engineVersion\":\"blink_2.2.4\",\"clusterId\":\"rcmp9x37ztfb63g1x7lt****\",\"queueName\":\"root.default\"}
```

正常返回示例

`XML` 格式

```
<requestId>B661F03F-0F9D-4EFF-A40F-4ED1519A549A</requestId>
```

`JSON` 格式

```
{
    "requestId": "B661F03F-0F9D-4EFF-A40F-4ED1519A549A"
}
```

7.4. DeleteJob

您可以通过DeleteJob OpenAPI删除已经下线的作业，删除未下线作业会报错。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
DELETE /api/v2/projects/[projectName]/jobs/[jobName] HTTPS
```

请求参数

名称	类型	是否必选	示例值	描述
jobName	String	是	job1	作业名称
projectName	String	是	project1	项目名称
RegionId	String	是	cn-hangzhou	区域ID **说明** 公共云用户请忽略此参数。

返回数据

名称	类型	示例值	描述
RequestId	String	78261AC6-B5C9-4514-B82F-39BCC45A5D63	请求ID

示例

请求示例

```
/api/v2/projects/project1/jobs/job1
```

正常返回示例

`XML` 格式

```
<RequestId>78261AC6-B5C9-4514-B82F-39BCC45A5D63</RequestId>
```

`JSON` 格式

```
{
 "RequestId": "78261AC6-B5C9-4514-B82F-39BCC45A5D63"
}
```

7.5. GetJob

您可以通过GetJob获取作业详细信息。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
GET /api/v2/projects/[projectName]/jobs/[jobName] HTTPS
```

请求参数

名称	类型	是否必选	示例值	描述
jobName	String	是	job1	作业名称
projectName	String	是	project1	项目名称
RegionId	String	否	cn-hangzhou	区域ID **说明** 公共云用户请忽略此参数。

返回数据

名称	类型	示例值	描述
Job	Struct		作业详情
ApiType	String	DATASTREAM	作业类型：DATASTREAM或SQL。
ClusterId	String	xxx	集群ID

名称	类型	示例值	描述
Code	String	xxxxx	运行代码： • SQL作业返回SQL代码。 • Datastream作业返回Datastream配置。
CreateTime	Long	1595493794000	作业创建时间
Creator	String	xxx	作业创建者
Description	String	test	作业备注信息
EngineVersion	String	blink-3.5.0-hotfix	引擎版本
FileId	String	75724	文件ID
FolderId	Long	26808	文件夹ID
IsCommitted	Boolean	true	是否为上线状态。
JobId	String	rd8r1lgccsnnchh9fqwukw6e	作业ID
JobName	String	datastream_enable	作业名称
JobType	String	FLINK_STREAM	作业类型： • FLINK_STREAM：流作业。 • FLINK_BATCH：批作业。
Modifier	String	xxx	最新修改者
ModifyTime	Long	1597743765000	最新修改时间
Packages	String	big-data-1.0-snapshot.jar	实例引用的包名称，多个包使用逗号（,）分隔，未引用为空。
PlanJson	String	{a:b}	作业上线的执行计划

名称	类型	示例值	描述
ProjectName	String	bayes_team	项目名称
Properties	String	k:v	作业运行配置参数
QueueName	String	root.default	队列名称
RequestId	String	2C6E2F4C-2FA1-4159-87DA-676DB15F4826	请求ID

示例

请求示例

```
/api/v2/projects/project1/jobs/job1
```

正常返回示例

`XML` 格式

```
<requestId>2C6E2F4C-2FA1-4159-87DA-676DB15F4826</requestId>
<job>
    <jobName>datastream_enable</jobName>
    <projectName>bayes_team</projectName>
    <jobType>FLINK_STREAM</jobType>
    <apiType>DATASTREAM</apiType>
    <code>xxxxx</code>
    <packages>big-data-1.0-snapshot.jar</packages>
    <isCommitted>true</isCommitted>
    <creator>xxx</creator>
    <createTime>1595493794000</createTime>
    <modifier>xxx</modifier>
    <modifyTime>1597743765000</modifyTime>
    <engineVersion>blink-3.5.0-hotfix</engineVersion>
    <clusterId>xxx</clusterId>
    <queueName>root.default</queueName>
    <folderId>26808</folderId>
    <jobId>rd8r1lgccsnnchh9fqwukw6e</jobId>
    <fileId>75724</fileId>
</job>
```

`JSON` 格式

```
{
    "requestId": "2C6E2F4C-2FA1-4159-87DA-676DB15F4826",
    "job": {
        "jobName": "datastream_enable",
        "projectName": "bayes_team",
        "jobType": "FLINK_STREAM",
        "apiType": "DATASTREAM",
        "code": "xxxxx",
        "packages": "big-data-1.0-snapshot.jar",
        "isCommitted": true,
        "creator": "xxx",
        "createTime": 1595493794000,
        "modifier": "xxx",
        "modifyTime": 1597743765000,
        "engineVersion": "blink-3.5.0-hotfix",
        "clusterId": "xxx",
        "queueName": "root.default",
        "folderId": 26808,
        "jobId": "rd8r1lgccsnnchh9fqwukw6e",
        "fileId": 75724
    }
}
```


7.6. ListJob

您可以通过ListJob OpenAPI获取指定project下满足条件的Job信息。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
GET /api/v2/projects/[projectName]/jobs HTTP/1.1
```

请求参数

名称	类型	位置	是否必选	示例值	描述
projectName	String	Path	是	project1	项目名称。
RegionId	String	Header	否	cn-hangzhou	区域ID。

名称	类型	位置	是否必选	示例值	描述
pageSize	Integer	Query	否	10	每页返回Job数。
pageIndex	Integer	Query	否	1	页码。
jobName	String	Query	否	job1	作业名称。
jobType	String	Query	否	FLINK_STREAM	作业种类： • FLINK_STREAM：流作业。 • FLINK_BATCH：批作业。
apiType	String	Query	否	SQL	API类型：DATASTREAM和SQL。
engineVersion	String	Query	否	blink_2.2.4	引擎版本。
clusterId	String	Query	否	d6wxwo5tnrmuamx2ly3m7vkz	集群ID。
queueName	String	Query	否	root.default	队列名称。
folderId	Long	Query	否	123	文件夹ID。
isShowFullField	Boolean	Query	否	true	123。

返回数据

名称	类型	示例值	描述
Jobs	Array of Job		作业详情。
Job			
ApiType	String	SQL	API类型：DATASTREAM和SQL两种。
ClusterId	String	d6wxwo5tnrmuamx2ly3m7vkz	集群ID。 **说明** 您可以使用listcluster获取集群ID信息。

名称	类型	示例值	描述
Code	String	code	作业代码。
CreateTime	Long	1548397575000	作业创建时间。
Creator	String	xxxx	创建者。
Description	String	test	作业备注描述。
EngineVersion	String	blink_2.2.4	引擎版本。
FolderId	Long	123	文件夹ID
IsCommitted	Boolean	true	作业是否已经提交运行。
JobId	String	123	Job ID。
JobName	String	job1	作业名称。
JobType	String	FLINK_STREAM	作业种类： • FLINK_STREAM：流作业。 • FLINK_BATCH：批作业。
Modifier	String	xxxx	最新修改者。
ModifyTime	Long	1548397575000	最新修改时间。
Packages	String	package1.jar	Package名称。
PlanJson	String	{a:b}	执行计划。
ProjectName	String	project1	项目名称。
Properties	String	k:v	作业运行配置。
QueueName	String	root.default	队列名称。

名称	类型	示例值	描述
PageIndex	Integer	1	页码。
PageSize	Integer	10	每页Job数。
RequestId	String	FD0FF8C0-779A-45EB-9674-FF3E127B10D2	请求ID。
TotalCount	Long	50	总Job数。
TotalPage	Integer	5	总页数。

示例

请求示例

```
/api/v2/projects/project1/jobs
```

正常返回示例

`XML` 格式

```
<RequestId>5A0AF0E2-A715-40FD-8A2B-1EDBF4213489</RequestId>
```

`JSON` 格式

```
{
    "RequestId": "5A0AF0E2-A715-40FD-8A2B-1EDBF4213489"
}
```

7.7. OfflineJob

您可以通过OfflineJob OpenAPI下线作业。

> **说明** 下线作业前，需要先停止运行作业。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
PUT /api/v2/projects/[projectName]/jobs/[jobName]/offline HTTPS
```

请求参数

名称	类型	是否必选	示例值	描述
jobName	String	是	job1	作业名称
projectName	String	是	project1	项目名称
RegionId	String	否	cn-hangzhou	区域ID **说明** 公共云用户请忽略此参数。

返回数据

名称	类型	示例值	描述
RequestId	String	5A0AF0E2-A715-40FD-8A2B-1EDBF4213489	请求ID

示例

请求示例

```
/api/v2/projects/project1/jobs/job1/offline
```

正常返回示例

`XML` 格式

```
<RequestId>5A0AF0E2-A715-40FD-8A2B-1EDBF4213489</RequestId>
```

`JSON` 格式

```
{
 "RequestId": "5A0AF0E2-A715-40FD-8A2B-1EDBF4213489"
}
```

7.8. StartJob

您可以通过StartJob OpenAPI启动指定项目下的目标作业。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
POST /api/v2/projects/[projectName]/jobs/[jobName]/instance HTTP/1.1
```

请求参数

名称	类型	位置	是否必选	示例值	描述
jobName	String	Path	是	job1	作业名称
projectName	String	Path	是	project1	项目名称
RegionId	String	Header	是	cn-hangzhou	区域ID
parameterJson	Json	Body	否	{"blink.checkpoint.interval.ms": "180000"}	JSON格式的配置参数，详情请参见配置参数表。

配置参数

参数名称	说明	备注
blink.checkpoint.mode	Checkpoint模式	AT_LEAST_ONCE：至少计算一次。 EXACTLY_ONCE：精确计算一次。
blink.checkpoint.interval.ms	Checkpoint间隔时间	数据类型为LONG，例如180000，单位为毫秒。
blink.checkpoint.timeout.ms	Checkpoint超时时间	数据类型为LONG，例如600000，单位为毫秒，默认值为10分钟。

参数名称	说明	备注
blink.job.option.jmMem MB	Job Manager内存	例如1024 MB。
blink.job.option	传给Flink Run的参数，可以传递多个参数，具体请参见Flink命令说明。	例如设置Task Manager的JVM线程的堆栈大小，可以配置：blink.job.option=-yD env.java.opts='-Xss8192K'。
client.jvm.option	编译阶段的JVM	例如：-Xss2048k。
blink.miniBatch.allowLatencyMs	Batch最大允许延迟时间	添加或删除该参数Job状态会丢失，修改值大小状态不会丢失，数据类型为LONG， 例如10000，单位为毫秒。
blink.miniBatch.size	Batch最大条数	添加或删除该参数Job状态会丢失，修改值大小状态不会丢失；此参数不能设置过大，如果一个Batch实际去完重的数据超过65536，在访问statebackend时可能触发JNI OOM异常；数据类型为LONG， 例如1000。
blink.microBatch.allowLatencyMs	Batch最大允许延迟时间，与minibatch的区别在于microbatch是基于batchmark触发的，而不是timer，具有更高的效率。	数据类型为LONG， 例如10000，Blink 2.2及以上版本支持该参数，单位为毫秒。
blink.localAgg.enabled	是否开启Local-Global Agg。需要开启MiniBatch，Local-Global Agg功能才生效。	添加或删除该参数Job状态会丢失，修改值大小状态不会丢失；Blink 2.2版本默认值为true，其他版本默认值为false。

参数名称	说明	备注
blink.partialAgg.enabled	是否开启Partial-Final Agg。Parital-Final Agg常用于解决COUNT DISTINCT热点问题。	默认值为false，Blink 2.2.0及以上版本支持该参数。
sql.exec.mini-batch.window.enabled	是否开启minibatch window	默认值为false（不开启），Blink 3.2.0及以上版本支持该参数。
sql.exec.source.idle.timeout.ms	Source空闲超时的时间，当source被标记为空闲后，下游就不会再等待该分区source发来的watermark，从而能触发window。	默认值为-1（不开启），取值大于0则表示开启。Blink 3.2.0及以上版本支持该参数。
blink.state.ttl.ms	TopN和groupBy节点的状态过期清理时间。状态在指定时间内没有被更新过则会被清除。	数据类型为LONG， 例如60000，单位为毫秒，默认为不清除。
blink.topn.cache.size	TopN缓存数据的条数。例如Top100，配置缓存10000条，即缓存了100组数据。	数据类型为LONG，默认值为10000。
blink.job.submit.timeoutInSeconds	SQL编译超时时间，建议设置最大不超过250s。	数据类型为LONG，默认值为180。
blink.job.timeZone	作业的时区	默认为Bayes前端时区，例如Asia/Shanghai。
blink.object.reuse	是否开启对象重用。如果不开启，chain一起的operator之前的数据传输会发生序列化或反序列化	Blink 2.2.0及以上版本默认值为true，其他版本默认值为false。

参数名称	说明	备注
blink.auto.watermark.interval.ms	Watermark发送的频率。需要权衡延迟和性能。太高会导致延迟增加，太低会导致频繁发送watermark，影响性能。	默认值为100ms。
blink.job.timestamp.reserve.ms	对于Ads、Petadata、SLS、Elasticsearch、DataHub类型的Sink，在写入timestamp类型时支持毫秒精度。	默认值为false，Blink 2.2.6及以上版本支持该参数。
yarn.app.blink.source.buffer-len	Connector source里面RecordReader线程作为生产者，上层ParallelReader线程作为消费者，中间的阻塞队列的长度，当数据源单条数据较小，数据量极大时，调高此项能提升部分性能。	默认值为10。
yarn.app.blink.source.source.idle-interval	Connector source里面ParallelReader无数据可读时，sleep的时间。	默认值为10，单位为100ms，Blink 2.2.5及以上版本支持该参数。
sql.exec.fault.tolerance.enabled	是否开启容错机制	默认值为false，Blink 3.4.0及以上版本支持该参数。
state.backend.niagara.ttl.ms	Niagara StateBackend中数据的保存时间	数据类型为LONG，例如129600000，默认值为一天半。
state.backend.block.cache.size.mb	Niagara StateBackend读缓存大小	默认值为256 MB。

参数名称	说明	备注
state.backend.mem.table.size.mb	Niagara StateBackend写缓存大小	默认值为128 MB。
state.backend.gemini.ttl.ms	Gemini StateBackend数据的保存时间	数据类型为LONG，例如129600000，默认值为一天半。
state.backend.gemini.mem.size.mb	Gemini StateBackend Native内存大小	默认值为384 MB。
state.backend.gemini.ttl.ms	Gemini StateBackend中数据的保存时间	数据类型为LONG，例如129600000，默认值为一天半。
state.backend.block.cache.size.mb	Gemini StateBackend读缓存大小	默认值为256 MB。
state.backend.mem.table.size.mb	Gemini StateBackend写缓存大小	默认值为128 MB。

返回数据

名称	类型	示例值	描述
RequestId	String	5313EF6E-6E3F-45DB-8E33-20DA279D8F3F	请求ID
instanceId	Long	415854	实例ID

示例

请求示例

```
/api/v2/projects/project1/jobs/job1/instance
```

正常返回示例

`XML` 格式

```
<instanceId>415854</instanceId>
<RequestId>5313EF6E-6E3F-45DB-8E33-20DA279D8F3F</RequestId>
```

JSON 格式

```
{
 "instanceId": 415854,
 "RequestId": "5313EF6E-6E3F-45DB-8E33-20DA279D8F3F"
}
```

7.9. UpdateJob

您可以通过UpdateJob OpenAPI更新作业属性。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
PUT /api/v2/projects/[projectName]/jobs/[jobName] HTTP/1.1
```

请求参数

名称	类型	位置	是否必选	示例值	描述
jobName	String	Path	是	job1	作业名称
projectName	String	Path	是	project1	项目名称
packages	String	Body	否	package1.jar	package名称
planJson	String	Body	否	{a:b}	作业执行计划
code	String	Body	否	code	作业代码
RegionId	String	Header	否	cn-hangzhou	区域ID
description	String	Body	否	test	作业备注
engineVersion	String	Body	否	blink_2.2.4	引擎版本

名称	类型	位置	是否必选	示例值	描述
clusterId	String	Body	否	d6wxwo5tnrmuamx2ly3m7vkz	集群ID **说明** 您可以使用listcluster获取集群ID信息。
queueName	String	Body	否	root.default	队列名称
folderId	Long	Body	否	123	文件夹ID
properties	String	Body	否	blink.checkpoint.interval.ms:180000	作业运行配置参数，参数详情请参见配置参数表。

配置参数

参数名称	说明	备注
blink.checkpoint.mode	Checkpoint模式	AT_LEAST_ONCE：至少计算一次。 EXACTLY_ONCE：精确计算一次。
blink.checkpoint.interval.ms	Checkpoint间隔时间	数据类型为LONG，例如180000，单位为毫秒。
blink.checkpoint.timeout.ms	Checkpoint超时时间	数据类型为LONG，例如600000，单位为毫秒，默认值为10分钟。
blink.job.option.jmMemMB	Job Manager内存	例如1024 MB。

参数名称	说明	备注
blink.job.option	传给Flink Run的参数，可以传递多个参数，具体请参见Flink命令说明。	例如设置Task Manager的JVM线程的堆栈大小，可以配置：blink.job.option=-yD env.java.opts='-Xss8192K'。
client.jvm.option	编译阶段的JVM	例如：-Xss2048k。
blink.miniBatch.allowLatencyMs	Batch最大允许延迟时间	添加或删除该参数Job状态会丢失，修改值大小状态不会丢失，数据类型为LONG，例如10000，单位为毫秒。
blink.miniBatch.size	Batch最大条数	添加或删除该参数Job状态会丢失，修改值大小状态不会丢失；此参数不能设置过大，如果一个Batch实际去完重的数据超过65536，在访问statebackend时可能触发JNI OOM异常；数据类型为LONG，例如1000。
blink.microBatch.allowLatencyMs	Batch最大允许延迟时间，与minibatch的区别在于microbatch是基于batchmark触发的，而不是timer，具有更高的效率。	数据类型为LONG，例如10000，Blink 2.2及以上版本支持该参数，单位为毫秒。
blink.localAgg.enabled	是否开启Local-Global Agg。需要开启MiniBatch，Local-Global Agg功能才生效。	添加或删除该参数Job状态会丢失，修改值大小状态不会丢失；Blink 2.2版本默认值为true，其他版本默认值为false。

参数名称	说明	备注
blink.partialAgg.enabled	是否开启Partial-Final Agg。Parital-Final Agg常用于解决COUNT DISTINCT热点问题。	默认值为false，Blink 2.2.0及以上版本支持该参数。
sql.exec.mini-batch.window.enabled	是否开启minibatch window	默认值为false（不开启），Blink 3.2.0及以上版本支持该参数。
sql.exec.source.idle.timeout.ms	Source空闲超时的时间，当source被标记为空闲后，下游就不会再等待该分区source发来的watermark，从而能触发window。	默认值为-1（不开启），取值大于0则表示开启。Blink 3.2.0及以上版本支持该参数。
blink.state.ttl.ms	TopN和groupBy节点的状态过期清理时间。状态在指定时间内没有被更新过则会被清除。	数据类型为LONG， 例如60000，单位为毫秒，默认为不清除。
blink.topn.cache.size	TopN缓存数据的条数。例如Top100，配置缓存10000条，即缓存了100组数据。	数据类型为LONG，默认值为10000。
blink.job.submit.timeoutInSeconds	SQL编译超时时间，建议设置最大不超过250s。	数据类型为LONG，默认值为180。
blink.job.timeZone	作业的时区	默认为Bayes前端时区，例如Asia/Shanghai。
blink.object.reuse	是否开启对象重用。如果不开启，chain一起的operator之前的数据传输会发生序列化或反序列化	Blink 2.2.0及以上版本默认值为true，其他版本默认值为false。
blink.auto.watermark.interval.ms	Watermark发送的频率。需要权衡延迟和性能。太高会导致延迟增加，太低会导致频繁发送watermark，影响性能。	默认值为100ms。

参数名称	说明	备注
blink.job.timestamp.reserve.ms	对于Ads、Petadata、SLS、Elasticsearch、DataHub类型的Sink，在写入timestamp类型时支持毫秒精度。	默认值为false，Blink 2.2.6及以上版本支持该参数。
yarn.app.blink.source.buffer-len	Connector source里面RecordReader线程作为生产者，上层ParallelReader线程作为消费者，中间的阻塞队列的长度，当数据源单条数据较小，数据量极大时，调高此项能提升部分性能。	默认值为10。
yarn.app.blink.source.source.idle-interval	Connector source里面ParallelReader无数据可读时，sleep的时间。	默认值为10，单位为100ms，Blink 2.2.5及以上版本支持该参数。
sql.exec.fault.tolerance.enabled	是否开启容错机制	默认值为false，Blink 3.4.0及以上版本支持该参数。
state.backend.niagara.ttl.ms	Niagara StateBackend中数据的保存时间	数据类型为LONG，例如129600000，默认值为一天半。
state.backend.block.cache.size.mb	Niagara StateBackend读缓存大小	默认值为256 MB。
state.backend.mem.table.size.mb	Niagara StateBackend写缓存大小	默认值为128 MB。
state.backend.gemini.ttl.ms	Gemini StateBackend数据的保存时间	数据类型为LONG，例如129600000，默认值为一天半。
state.backend.gemini.mem.size.mb	Gemini StateBackend Native内存大小	默认值为384 MB。

参数名称	说明	备注
state.backend.gemini.ttl.ms	Gemini StateBackend中数据的保存时间	数据类型为LONG，例如129600000，默认值为一天半。
state.backend.block.cache.size.mb	Gemini StateBackend读缓存大小	默认值为256 MB。
state.backend.mem.table.size.mb	Gemini StateBackend写缓存大小	默认值为128 MB。

返回数据

名称	类型	示例值	描述
RequestId	String	C3FC248C-1CF4-4A64-97C1-0BD667A4D09D	请求ID

示例

请求示例

```
/api/v2/projects/project1/jobs/job1
```

正常返回示例

`XML` 格式

```
<RequestId>C3FC248C-1CF4-4A64-97C1-0BD667A4D09D</RequestId>
```

`JSON` 格式

```
{
 "RequestId": "C3FC248C-1CF4-4A64-97C1-0BD667A4D09D"
}
```

7.10. ValidateJob

您可以通过ValidateJob OpenAPI对作业进行语法检查。如果检查通过，则返回参数。如果检查不通过，则会返回解析SQL出现错误的报错信息。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
GET /api/v2/projects/[projectName]/jobs/[jobName]/validate HTTPS
```

请求参数

名称	类型	位置	是否必选	示例值	描述
jobName	String	Path	是	job1	作业名称
projectName	String	Path	是	project1	项目名称
RegionId	String	Header	否	cn-hangzhou	区域ID

返回数据

名称	类型	示例值	描述
JobInOut	Struct		作业输入输出
Dims	Array of Dim		作业中的维表
Dim			
Alias	String	phoneNumber	维表别名
Name	String	phoneNumber	表名
Properties	Map	k:v	配置组
Type	String	rds	表类型
Workspace	String	project1	表所属项目
Inputs	Array of Input		输入数组详情
Input			

名称	类型	示例值	描述
Alias	String	datahub_input1	源表别名
Name	String	datahub_input1	表名
Properties	Map	k:v	配置组
Type	String	datahub	表类型
Workspace	String	project1	表所属项目
Outputs	Array of Output		返回数组详情
Output			
Alias	String	result_infor	结果表别名
Name	String	result_infor	表名
Properties	Map	k:v	配置组
Type	String	rds	表类型
Workspace	String	project1	表所属项目
RequestId	String	5A3EB3CC-D899-4B0B-AF1F-1C0C0B6A0F91	请求ID

示例

请求示例

```
/api/v2/projects/project1/jobs/job1/validate
```

正常返回示例

`XML` 格式

```
<JobInOut>
    <Outputs>
        <Output>
            <Type>rds</Type>
            <Alias>result_infor</Alias>
            <Properties>
                <type>rds</type>
            </Properties>
            <Workspace/>
        </Output>
    </Outputs>
    <Dims>
        <Dim>
            <Type>ots</Type>
            <Alias>phoneNumber</Alias>
            <Properties>
                <type>ots</type>
            </Properties>
        </Dim>
    </Dims>
    <Inputs>
        <Input>
            <Type>datahub</Type>
            <Alias>datahub_input1</Alias>
            <Properties>
                <type>datahub</type>
            </Properties>
        </Input>
    </Inputs>
</JobInOut>
<RequestId>5A3EB3CC-D899-4B0B-AF1F-1C0C0B6A0F91</RequestId>
```

`JSON` 格式

```
{
 "JobInOut": {
  "Outputs": {
   "Output": [
    {
     "Type": "rds",
     "Alias": "result_infor",
     "Properties": {
      "type": "rds"
     },
     "Workspace": ""
    }
   ]
  },
  "Dims": {
   "Dim": [
    {
     "Type": "ots",
     "Alias": "phoneNumber",
     "Properties": {
      "type": "ots"
     }
    }
   ]
  },
  "Inputs": {
   "Input": [
    {
     "Type": "datahub",
     "Alias": "datahub_input1",
     "Properties": {
      "type": "datahub"
     }
    }
   ]
  }
 },
 "RequestId": "5A3EB3CC-D899-4B0B-AF1F-1C0C0B6A0F91"
}
```


7.11. CalcPlanJsonResource

您可以通过CalcPlanJsonResource获取作业执行计划所需要的CPU、内存等资源消耗信息。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
GET /api/v2/projects/[projectName]/jobs/[jobName]/planjson-resource HTTP/1.1
```

请求参数

名称	类型	位置	是否必选	示例值	描述
jobName	String	Path	是	job1	作业名称。
projectName	String	Path	是	project1	项目名称。
RegionId	String	Header	否	cn-shanghai	地域ID。

返回数据

名称	类型	示例值	描述
PlanJsonResource	Struct		执行计划的资源信息。
Cores	Float	5	作业所消耗的CPU内核个数。
MemoryGB	Float	12.5	作业所消耗的内存大小，单位是GB。
RequestId	String	4D3D89FC-4C0F-4FF5-82BD-BF8C00F22129	请求ID。

示例

请求示例

```
/api/v2/projects/project1/jobs/job1/planjson-resource
```

正常返回示例

`XML` 格式

```
<PlanJsonResource>
    <Cores>5</Cores>
    <MemoryGB>12.5</MemoryGB>
</PlanJsonResource>
<RequestId>4D3D89FC-4C0F-4FF5-82BD-BF8C00F22129</RequestId>
```

`JSON` 格式

```
{
 "PlanJsonResource": {
  "Cores": 5,
  "MemoryGB": 12.5
 },
 "RequestId": "4D3D89FC-4C0F-4FF5-82BD-BF8C00F22129"
}
```

7.12. UpdateAutoScaleConfig

您可以通过本API对开启AutoScale功能的作业进行AutoScale功能的关闭或者关闭后的再次开启。

UpdateAutoScaleConfig仅适用于使用AutoScale功能启动的作业，未使用AutoScale功能的作业不支持该API。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

除公共请求头，该接口还需要指定以下请求头。关于公共请求头，请参见公共请求参数文档。

名称	类型	是否必选	示例	描述
RegionId	String	否	cn-shanghai	地域ID

请求语法

```
PUT /api/v2/projects/[projectName]/jobs/[jobName]/instance/[instanceId]/autoscale/config HTTP/1.1
```

请求参数

名称	类型	是否必选	示例值	描述
configJson	Json	是	{"autoscale.enable":"true"}	设置开启或者关闭。
instanceId	Long	是	-1	作业实例ID。流作业填 `-1` ，批作业暂时不支持该API。
jobName	String	是	job1	作业名称。
projectName	String	是	project1	项目名称。

返回数据

名称	类型	示例值	描述
RequestId	String	A017B343-33DC-4A97-BAC5-4B57872D5B61	请求ID

示例

请求示例

```
PUT /api/v2/projects/[projectName]/jobs/[jobName]/instance/[instanceId]/autoscale/config HTTP/1.1
RegionId:
公共请求头
```

正常返回示例

`XML` 格式

```
<RequestId>A017B343-33DC-4A97-BAC5-4B57872D5B61</RequestId>
<Metrics>
    <metricName>et2hunbu.yarn.resourcemanager.jvmmetrics.MemHeapUsed</metricName>
    <dps>
        <1574244620>3675.839599609375</1574244620>
        <1574244640>3817.712646484375</1574244640>
        <1574244660>3864.84130859375</1574244660>
        <1574244680>3973.963134765625</1574244680>
        <1574244700>4121.57958984375</1574244700>
        <1574244720>4170.6123046875</1574244720>
        <1574244740>4279.62109375</1574244740>
        <1574244760>4427.05615234375</1574244760>
        <1574244780>4472.11474609375</1574244780>
    </dps>
    <summary>6700.7467978583445</summary>
    <tags/>
</Metrics>
```

`JSON` 格式

```
{
 "Metrics":[
  {
   "tags":{},
   "summary":6700.7467978583445,
   "dps":{
    "1574244640":3817.712646484375,
    "1574244720":4170.6123046875,
    "1574244780":4472.11474609375,
    "1574244740":4279.62109375,
    "1574244680":3973.963134765625,
    "1574244760":4427.05615234375,
    "1574244700":4121.57958984375,
    "1574244660":3864.84130859375,
    "1574244620":3675.839599609375
   },
   "metricName":"et2hunbu.yarn.resourcemanager.jvmmetrics.MemHeapUsed"
  }
 ],
 "RequestId":"A017B343-33DC-4A97-BAC5-4B57872D5B61"
}
```


错误码

访问错误中心查看更多错误码。

7.13. GetJobLatestAutoScalePlan

您可以通过GetJobLatestAutoScalePlan获取作业最新的自动调优执行计划。

> **说明** 仅开启自动调优功能的作业支持GetJobLatestAutoScalePlan。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
GET /api/v2/projects/[projectName]/jobs/[jobName]/autoscale/latestplanjson HTTP/1.1
```

请求参数

名称	类型	位置	是否必选	示例值	描述

名称	类型	位置	是否必选	示例值	描述
jobName	String	Path	是	job1	作业名
projectName	String	Path	是	project1	项目名
RegionId	String	Header	是	cn-shanghai	地区ID。

返回数据

名称	类型	示例值	描述
PlanJson	String	{k:v}	作业执行计划
RequestId	String	C0D76C25-2056-43E4-B4EA-A7BBF774BAC1	请求ID

示例

请求示例

```
/api/v2/projects/project1/jobs/job1/autoscale/latestplanjson
```

正常返回示例

`XML` 格式

```
<RequestId>C0D76C25-2056-43E4-B4EA-A7BBF774BAC1</RequestId>
```

`JSON` 格式

```
{
 "RequestId": "C0D76C25-2056-43E4-B4EA-A7BBF774BAC1"
}
```

7.14. GetRawPlanJson

您可以通过GetRawPlanJson OpenAPI获取作业初始执行计划。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
GET /api/v2/projects/[projectName]/jobs/[jobName]/planjson HTTP/1.1
```

请求参数

名称	类型	位置	是否必选	示例值	描述
jobName	String	Path	是	job1	作业名称。
projectName	String	Path	是	project1	项目名称。
RegionId	String	Header	否	cn-hangzhou	区域ID。
autoconfEnable	Boolean	Query	否	true	是否开启智能调优，取值如下： • true：开启，根据作业历史Metric，生成一份执行计划。 • false：不开启，使用默认执行计划。
expectedCore	Float	Query	否	1	期望CPU数。
expectedGB	Float	Query	否	4	期望内存。
AdvisorAction	String	Query	否	advancedInfo	取值如下： • advancedInfo：获取作业plan的同时，也获取作业的SQL建议。 • advisor：仅获取作业的SQL建议。 **说明** 仅Blink 3.7.0以上版本支持该参数。

返回数据

名称	类型	示例值	描述
RequestId	String	9B8DCCAB-6C1C-4195-8847-F17C3A36007E	请求ID。

名称	类型	示例值	描述
SessionId	String	6sozg4hlznp8ylrgp6dpoxpisewil57ymeps4e0ues7ikwt2nz12neu7w5u28dt2066o9dn6kzzhzjtqshgbbl6uwosbq9g930ucmnzopviyzqgefhpyrpwtbly5r1wg4vtsd8k8ud4ukhuiu39ce	会话ID，需要传入checkrowplanjson接口中使用。

示例

请求示例

```
/api/v2/projects/project1/jobs/job1/planjson
```

正常返回示例

`XML` 格式

```
<RequestId>9B8DCCAB-6C1C-4195-8847-F17C3A36007E</RequestId>
<SessionId>6sozg4hlznp8ylrgp6dpoxpisewil57ymeps4e0ues7ikwt2nz12neu7w5u28dt2066o9dn6kzzhzjtqshgbbl6uwosbq9g930ucmnzopviyzqgefhpyrpwtbly5r1wg4vtsd8k8ud4ukhuiu39ce</SessionId>
```

`JSON` 格式

```
{
 "RequestId": "9B8DCCAB-6C1C-4195-8847-F17C3A36007E",
 "SessionId": "6sozg4hlznp8ylrgp6dpoxpisewil57ymeps4e0ues7ikwt2nz12neu7w5u28dt2066o9dn6kzzhzjtqshgbbl6uwosbq9g930ucmnzopviyzqgefhpyrpwtbly5r1wg4vtsd8k8ud4ukhuiu39ce"
}
```

8.Package

8.1. CreatePackage

您可以通过CreatePackage OpenAPI获取您在OSS中创建的Package，并将该Package传送到作业的执行节点。

> 说明
> - Package的版本信息由您自行维护，上传时可以选填MD5值，开发控制台协助校验。
> - 共享集群用户无法使用CreatePackage。
> - 独享集群用户只能使用创建集群时的OSS信息，填写时只需要填写ossPath，无需填写完整信息。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
POST /api/v2/projects/[projectName]/packages HTTP/1.1
```

请求参数

名称	类型	位置	是否必选	示例值	描述
ossBucket	String	Body	是	blinktest2.oss-cn-hangzhou-internal.aliyuncs.com	OSSBucket
ossEndpoint	String	Body	是	oss-cn-hangzhou-internal.aliyuncs.com	OSS接入点
ossOwner	String	Body	是	user1	OSS所有者
ossPath	String	Body	是	path1/path2/a.jar	Package在OSS中的路径
packageName	String	Body	是	package1.jar	Package名称

名称	类型	位置	是否必选	示例值	描述
projectName	String	Path	是	project1	项目名称
type	String	Body	是	JAR	Package类型： • JAR：JAR包 • DICTIONARY：普通文件 • SCRIPT：脚本 • PYTHON：Python文件或者ZIP包
originName	String	Body	否	12222	Package原名
description	String	Body	否	test	Package的备注描述
md5	String	Body	否	3F7468C153E529B141C326332DF15D05	Package的MD5值
tag	String	Body	否	{"function":"group by time"}	Package的标记
RegionId	String	Header	否	cn-hangzhou	区域ID **说明** 公共云用户请忽略此参数。

返回数据

名称	类型	示例值	描述
RequestId	String	F85A9751-638B-4186-B5E5-3F66D8A1CBFE	请求ID

示例

请求示例

```
/api/v2/projects/project1/packages
{\"packageName\":\"package1.jar\",\"type\":\"JAR\",\"ossEndpoint\":\"oss-cn-hangzhou-internal.aliyuncs.com\",\"ossBucket\":\"blinktest2.oss-cn-hangzhou-internal.aliyuncs.com\",\"ossOwner\":\"user1\",\"ossPath\":\"path1/path2/a.jar\"}
```

正常返回示例

`XML` 格式

```
<RequestId>F85A9751-638B-4186-B5E5-3F66D8A1CBFE</RequestId>
```

`JSON` 格式

```
{
 "RequestId": "F85A9751-638B-4186-B5E5-3F66D8A1CBFE"
}
```

8.2. DeletePackage

您可以通过DeletePackag OpenAPI删除开发控制台上通过CreatePackage创建的Package。删除后开发控制台无法再获取对应的Package信息。

> **说明** 使用DeletePackage删除不存在的Package时，系统会报错。如果已上线的作业已引用该Package，删除该Package时也会报错。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
DELETE /api/v2/projects/[projectName]/packages/[packageName] HTTPS
```

请求参数

名称	类型	是否必选	示例值	描述
packageName	String	是	package.jar	Package名称
projectName	String	是	project1	项目名称
RegionId	String	是	cn-hangzhou	区域ID **说明** 公共云用户请忽略此参数。

返回数据

名称	类型	示例值	描述
RequestId	String	A604B9FA-D181-43D7-A064-5FF971B90466	请求ID

示例

请求示例

```
/api/v2/projects/project1/packages/package.jar
```

正常返回示例

`XML` 格式

```
<RequestId>A604B9FA-D181-43D7-A064-5FF971B90466</RequestId>
```

`JSON` 格式

```
{
 "RequestId": "A604B9FA-D181-43D7-A064-5FF971B90466"
}
```

8.3. GetPackage

您可以通过GetPackage OpenAPI获取Package详细信息。

> **说明** GetPackage只能获取到通过Web或CreatePackage接口在开发控制台上保存的Package相关信息。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
GET /api/v2/projects/[projectName]/packages/[packageName] HTTP/1.1
```

请求参数

名称	类型	位置	是否必选	示例值	描述
packageName	String	Path	是	aliyun-java-sdk-foas-2.9.0.jar	Package名称。
projectName	String	Path	是	project1	项目名称。
RegionId	String	Header	是	cn-hangzhou	区域ID **说明** 公共云用户请忽略此参数。

返回数据

名称	类型	示例值	描述
Package	Struct		Package详情。
CreateTime	Long	1600156643000	创建时间（13位时间戳，精确到毫秒）。
Creator	String	1709064687573327	创建者。
Description	String	adas	Package备注。
Md5	String	e0899e291f0524c6e6572c126599f73b	Package的Md5值。
Modifier	String	1709064687573327	最新修改者。
ModifyTime	Long	1600156643000	最新修改时间。
OriginName	String	aliyun-java-sdk-foas-2.9.0.jar	Package别名。
OssBucket	String	blinktest2.oss-cn-hangzhou-internal.aliyuncs.com	OSS Bucket。
OssEndpoint	String	oss-cn-hangzhou-internal.aliyuncs.com	OSS接入点。

名称	类型	示例值	描述
OssOwner	String	owner1	OSS所有者。
OssPath	String	79zec7svps7hca0xnogbqpvu/aliyun-java-sdk-foas-2.9.0.jar/l1jsjfzlps4z4pk5rhutb91k.bayes	OSS路径。
PackageName	String	package1.jar	Package名称。
ProjectName	String	project1	项目名称。
ScanErrorMessage	String	未开始进行安全检查。	JAR包安全扫描信息。
ScanExtBizNo	String	5467**	安全扫描ID。
ScanLink	String	http://xxxxx	扫描详情URI。
ScanState	String	NOT_START	JAR包安全扫描状态。参数取值如下： • SKIP_SCAN：无需进行安全检查。 • NOT_START：未开始进行安全检查。 • EXECUTING：正在进行安全检查。 • PASS：安全检查通过。 • FAIL：安全检查未通过。 • EXCEPTION：安全检查出错。
Tag	String	{"function":"group by time"}	Package的标记。
Type	String	JAR	Package类型： • JAR：JAR包 • DICTIONARY：普通文件 • SCRIPT：脚本 • PYTHON：Python文件或者ZIP包
RequestId	String	7076AAD2-EF55-479A-867E-3245B9105E1D	请求ID。

示例

请求示例

```
/api/v2/projects/project1/packages/aliyun-java-sdk-foas-2.9.0.jar
```

正常返回示例

`XML` 格式

```
<RequestId>7076AAD2-EF55-479A-867E-3245B9105E1D</RequestId>
<Package>
    <ModifyTime>1600156643000</ModifyTime>
    <Description>adas</Description>
    <ProjectName>project1</ProjectName>
    <PackageName>aliyun-java-sdk-foas-2.9.0.jar</PackageName>
    <CreateTime>1600156643000</CreateTime>
    <Creator>1709064687573327</Creator>
    <OriginName>aliyun-java-sdk-foas-2.9.0.jar</OriginName>
    <Type>JAR</Type>
    <OssPath>79zec7svps7hca0xnogbqpvu/aliyun-java-sdk-foas-2.9.0.jar/l1jsjfzlps4z4pk5rhutb9
1k.bayes</OssPath>
    <Modifier>1709064687573327</Modifier>
    <Md5>e0899e291f0524c6e6572c126599f73b</Md5>
</Package>
```

`JSON` 格式

```
{
 "RequestId": "7076AAD2-EF55-479A-867E-3245B9105E1D",
 "Package": {
  "ModifyTime": 1600156643000,
  "Description": "adas",
  "ProjectName": "project1",
  "PackageName": "aliyun-java-sdk-foas-2.9.0.jar",
  "CreateTime": 1600156643000,
  "Creator": "1709064687573327",
  "OriginName": "aliyun-java-sdk-foas-2.9.0.jar",
  "Type": "JAR",
  "OssPath": "79zec7svps7hca0xnogbqpvu/aliyun-java-sdk-foas-2.9.0.jar/l1jsjfzlps4z4pk5rhutb
91k.bayes",
  "Modifier": "1709064687573327",
  "Md5": "e0899e291f0524c6e6572c126599f73b"
 }
}
```

8.4. GetRefPackageJob

您可以通过GetRefPackageJob OpenAPI获取指定Package的所有作业详情。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
GET /api/v2/projects/[projectName]/packages/[packageName]/jobs HTTPS
```

请求参数

名称	类型	是否必选	示例值	描述
packageName	String	是	package1.jar	Package名称
projectName	String	是	project1	项目名称
RegionId	String	否	cn-hangzhou	区域ID **说明** 公共云用户请忽略此参数。
pageSize	Integer	否	50	包大小
pageIndex	Integer	否	123	索引

返回数据

名称	类型	示例值	描述
Jobs	Array of Job		作业详情
Job			
ApiType	String	DATASTREAM	API类型：DATASTREAM或SQL
ClusterId	String	et2cloud****	集群ID
Code	String	blink.main.class=com.hjc.test.blink_test.ChinanTopSpeedWindowing2\r\nblink.job.name=datastream_prestest	作业代码

名称	类型	示例值	描述
CreateTime	Long	1551161842000	作业创建时间
Creator	String	1709064687573327	作业创建者
Description	String	test	作业备注
EngineVersion	String	blink_2.2.4	引擎版本
FolderId	Long	10148	文件夹ID
IsCommitted	Boolean	true	是否上线
JobId	String	9t9m6jfw8rb44iexhbpv3tnz	作业ID
JobName	String	job1	作业名称
JobType	String	FLINK_STREAM	作业类型： • FLINK_STREAM：流作业 • FLINK_BATCH：批作业
Modifier	String	1709064687573327	最新修改者
ModifyTime	Long	1548397575000	最新修改时间
Packages	String	blink_test_datastream.jar	Package名称，多个Package请使用逗号（,）分隔。
		{\n \"nodes\": [\n {\n \"id\": 1,\n \"type\": \"Source: Custom Source\",\n \"pact\": \"Data Source\",\n \"contents\": \"Source: Custom Source\",\n \"parallelism\": 1\n },\n {\n \"id\": 2,\n \"type\": \"Timestamps/Watermarks\",\n	

名称	类型	示例值	描述
PlanJson	String	\"pact\": \"Operator\",\n \"contents\": \"Timestamps/Watermarks\",\n \"parallelism\": 1,\n \"predecessors\": [\n {\n \"id\": 1,\n \"ship_strategy\": \"FORWARD\",\n \"side\": \"second\"\n }\n]\n },\n {\n \"id\": 4,\n \"type\": \"TriggerWindow(GlobalWindows(), ListStateDescriptor{serializer=org.apache.flink.api.common.typeutils.base.ListSerializer@b2107f67}, DeltaTrigger(com.hjc.test.blink_test.ChinanTopSpeedWindowing$1@350aac89, 50.0), TimeEvictor(10000), WindowedStream.reduce(WindowedStream.java:258))\",\n \"pact\": \"Operator\",\n \"contents\": \"TriggerWindow(GlobalWindows(), ListStateDescriptor{serializer=org.apache.flink.api.common.typeutils.base.ListSerializer@b2107f67}, DeltaTrigger(com.hjc.test.blink_test.ChinanTopSpeedWindowing$1@350aac89, 50.0), TimeEvictor(10000), WindowedStream.reduce(WindowedStream.java:258))\",\n \"parallelism\": 1,\n \"predecessors\": [\n {\n \"id\": 2,\n \"ship_strategy\": \"KEY-GROUP\",\n \"side\": \"second\"\n }\n]\n	作业上线的执行计划

名称	类型	示例值	描述
		},\n {\n \"id\": 5,\n \"type\": \"Sink: Unnamed\",\n \"pact\": \"Data Sink\",\n \"contents\": \"Sink: Unnamed\",\n \"parallelism\": 1,\n \"predecessors\": [\n {\n \"id\": 4,\n \"ship_strategy\": \"FORWARD\",\n \"side\": \"second\"\n }\n]\n }\n]\n}	
ProjectName	String	project1	项目名称
Properties	String	#checkpoint模式：EXACTLY_ONCE 或者 AT_LEAST_ONCE\n#blink.checkpoint.mode=EXACTLY_ONCE\n\n#checkpoint间隔时间，单位毫秒\n#blink.checkpoint.interval.ms=180000\n\n#rocksdb的数据生命周期，单位毫秒\n#state.backend.rocksdb.ttl.ms=129600000\n	作业运行参数配置
QueueName	String	root.bayes_team	队列名称
PageIndex	Integer	1	返回第几页
PageSize	Integer	10	每页包含的作业数
RequestId	String	31FDAE27-F135-4DAB-9C75-FADD6B0C4D1D	请求ID
TotalCount	Long	1	作业总数
TotalPage	Integer	1	页码总数

示例

请求示例

```
/api/v2/projects/project1/packages/package1.jar/jobs
```

正常返回示例

`XML` 格式

```
<TotalCount>1</TotalCount>
<TotalPage>1</TotalPage>
<PageSize>10</PageSize>
<RequestId>31FDAE27-F135-4DAB-9C75-FADD6B0C4D1D</RequestId>
<Jobs>
    <Job>
        <ModifyTime>1583722910000</ModifyTime>
        <Description/>
        <EngineVersion>blink-3.4.4</EngineVersion>
        <ProjectName>bayes_team</ProjectName>
        <ClusterId>et2cloud****</ClusterId>
        <PlanJson>{
  "nodes": [
    {
      "id": 1,
      "type": "Source: Custom Source",
      "pact": "Data Source",
      "contents": "Source: Custom Source",
      "parallelism": 1
    },
    {
      "id": 2,
      "type": "Timestamps/Watermarks",
      "pact": "Operator",
      "contents": "Timestamps/Watermarks",
      "parallelism": 1,
      "predecessors": [
        {
          "id": 1,
          "ship_strategy": "FORWARD",
          "side": "second"
        }
      ]
    },
    {
      "id": 4,
      "type": "TriggerWindow(GlobalWindows(), ListStateDescriptor{serializer=org.apache.flink.api.common.typeutils.base.ListSerializer@b2107f67}, DeltaTrigger(com.hjc.test.blink_test.ChinanTopSpeedWindowing$1@350aac89, 50.0), TimeEvictor(10000), WindowedStream.reduce(WindowedStream.java:258))",
      "pact": "Operator",
      "contents": "TriggerWindow(GlobalWindows(), ListStateDescriptor{serializer=org.apache.flink.api.common.typeutils.base.ListSerializer@b2107f67}, DeltaTrigger(com.hjc.test.blink_test.ChinanTopSpeedWindowing$1@350aac89, 50.0), TimeEvictor(10000), WindowedStream.reduce(WindowedStream.java:258))",
      "parallelism": 1,
```

```
      "predecessors": [
        {
          "id": 2,
          "ship_strategy": "KEY-GROUP",
          "side": "second"
        }
      ]
    },
    {
      "id": 5,
      "type": "Sink: Unnamed",
      "pact": "Data Sink",
      "contents": "Sink: Unnamed",
      "parallelism": 1,
      "predecessors": [
        {
          "id": 4,
          "ship_strategy": "FORWARD",
          "side": "second"
        }
      ]
    }
  ]
}</PlanJson>
        <CreateTime>1551161842000</CreateTime>
        <JobName>job1</JobName>
        <Creator>1709064687573327</Creator>
        <FolderId>10148</FolderId>
        <Properties>#checkpoint模式：EXACTLY_ONCE 或者 AT_LEAST_ONCE
#blink.checkpoint.mode=EXACTLY_ONCE
#checkpoint间隔时间，单位毫秒
#blink.checkpoint.interval.ms=180000
#rocksdb的数据生命周期，单位毫秒
#state.backend.rocksdb.ttl.ms=129600000
</Properties>
        <Code>blink.main.class=com.hjc.test.blink_test.ChinanTopSpeedWindowing2
blink.job.name=datastream_prestest</Code>
        <JobType>FLINK_STREAM</JobType>
        <Packages>blink_test_datastream.jar</Packages>
        <ApiType>DATASTREAM</ApiType>
        <IsCommitted>true</IsCommitted>
        <Modifier>1709064687573327</Modifier>
        <QueueName>root.bayes_team</QueueName>
        <JobId>9t9m6jfw8rb44iexhbpv3tnz</JobId>
    </Job>
</Jobs>
<PageIndex>1</PageIndex>
```

`JSON` 格式

```
{
 "TotalCount": 1,
 "TotalPage": 1,
 "PageSize": 10,
```

```
 "RequestId": "31FDAE27-F135-4DAB-9C75-FADD6B0C4D1D",
 "Jobs": {
  "Job": [
   {
    "ModifyTime": 1583722910000,
    "Description": "",
    "EngineVersion": "blink-3.4.4",
    "ProjectName": "bayes_team",
    "ClusterId": "et2cloud****",
    "PlanJson": "{\n  \"nodes\": [\n    {\n      \"id\": 1,\n      \"type\": \"Source: Cust
om Source\",\n      \"pact\": \"Data Source\",\n      \"contents\": \"Source: Custom Source
\",\n      \"parallelism\": 1\n    },\n    {\n      \"id\": 2,\n      \"type\": \"Timestamp
s/Watermarks\",\n      \"pact\": \"Operator\",\n      \"contents\": \"Timestamps/Watermarks
\",\n      \"parallelism\": 1,\n      \"predecessors\": [\n        {\n          \"id\": 1,\
n          \"ship_strategy\": \"FORWARD\",\n          \"side\": \"second\"\n        }\n
]\n    },\n    {\n      \"id\": 4,\n      \"type\": \"TriggerWindow(GlobalWindows(), ListSt
ateDescriptor{serializer=org.apache.flink.api.common.typeutils.base.ListSerializer@b2107f67
}, DeltaTrigger(com.hjc.test.blink_test.ChinanTopSpeedWindowing$1@350aac89, 50.0), TimeEvic
tor(10000), WindowedStream.reduce(WindowedStream.java:258))\",\n      \"pact\": \"Operator\
",\n      \"contents\": \"TriggerWindow(GlobalWindows(), ListStateDescriptor{serializer=org
.apache.flink.api.common.typeutils.base.ListSerializer@b2107f67}, DeltaTrigger(com.hjc.test
.blink_test.ChinanTopSpeedWindowing$1@350aac89, 50.0), TimeEvictor(10000), WindowedStream.r
educe(WindowedStream.java:258))\",\n      \"parallelism\": 1,\n      \"predecessors\": [\n
{\n          \"id\": 2,\n          \"ship_strategy\": \"KEY-GROUP\",\n          \"side\": \
"second\"\n        }\n      ]\n    },\n    {\n      \"id\": 5,\n      \"type\": \"Sink: Unn
amed\",\n      \"pact\": \"Data Sink\",\n      \"contents\": \"Sink: Unnamed\",\n      \"pa
rallelism\": 1,\n      \"predecessors\": [\n        {\n          \"id\": 4,\n          \"sh
ip_strategy\": \"FORWARD\",\n          \"side\": \"second\"\n        }\n      ]\n    }\n  ]
\n}",
    "CreateTime": 1551161842000,
    "JobName": "job1",
    "Creator": "1709064687573327",
    "FolderId": 10148,
    "Properties": "#checkpoint模式：EXACTLY_ONCE 或者 AT_LEAST_ONCE\n#blink.checkpoint.mode=
EXACTLY_ONCE\n\n#checkpoint间隔时间，单位毫秒\n#blink.checkpoint.interval.ms=180000\n\n#rocks
db的数据生命周期，单位毫秒\n#state.backend.rocksdb.ttl.ms=129600000\n",
    "Code": "blink.main.class=com.hjc.test.blink_test.ChinanTopSpeedWindowing2\r\nblink.job
.name=datastream_prestest",
    "JobType": "FLINK_STREAM",
    "Packages": "blink_test_datastream.jar",
    "ApiType": "DATASTREAM",
    "IsCommitted": true,
    "Modifier": "1709064687573327",
    "QueueName": "root.bayes_team",
    "JobId": "9t9m6jfw8rb44iexhbpv3tnz"
   }
  ]
 },
 "PageIndex": 1
}
```


8.5. ListPackage

您可以通过ListPackag OpenAPI获取指定Project下满足特定条件的Package信息。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
GET /api/v2/projects/[projectName]/packages HTTPS
```

请求参数

名称	类型	是否必选	示例值	描述
projectName	String	是	project1	项目名称
packageName	String	否	package1.jar	Package名称
type	String	否	JAR	Package类型： • JAR：JAR包 • DICTIONARY：普通文件 • SCRIPT：脚本 • PYTHON：Python文件或者ZIP包
pageSize	Integer	否	10	每页包含Package数量
pageIndex	Integer	否	1	索引
tag	String	否	{"function":"group by time"}	Package的标记
RegionId	String	否	cn-hangzhou	区域ID **说明** 公共云用户请忽略此参数。

返回数据

名称	类型	示例值	描述
Packages	Array of Package		Package详情
Package			
CreateTime	Long	1600156643000	创建时间（13位时间戳，精确到毫秒）
Creator	String	1709064687573327	创建者
Description	String	adas	Package备注
Md5	String	e0899e291f0524c6e6572c126599f73b	Package的Md5值
Modifier	String	1709064687573327	最新修改者
ModifyTime	Long	1600156643000	最新修改时间
OriginName	String	aliyun-java-sdk-foas-2.9.0.jar	Package别名
OssBucket	String	79zec7svps7hca0xnogbqpvu/aliyun-java-sdk-foas-2.9.0.jar/l1jsjfzlps4z4pk5rhutb91k.bayes	OssBucket
OssEndpoint	String	oss-cn-hangzhou-internal.aliyuncs.com	Oss接入点
OssOwner	String	owner1	Oss拥有者
OssPath	String	path1/path2/a.jar	Oss路径
PackageName	String	aliyun-java-sdk-foas-2.9.0.jar	Package名称
ProjectName	String	project1	项目名称
Tag	String	{"function":"group by time"}	Package的标记

名称	类型	示例值	描述
Type	String	JAR	Package 类型： • JAR：JAR包 • DICTIONARY：普通文件 • SCRIPT：脚本 • PYTHON：Python文件或者ZIP包
PageIndex	Integer	1	第几页
PageSize	Integer	10	每页包含Package数量
RequestId	String	F33B93C0-56E6-43BD-B30F-E1FF1D4EF1EE	请求ID
TotalCount	Long	4	Package总数
TotalPage	Integer	1	总页面数

示例

请求示例

```
/api/v2/projects/project1/packages
```

正常返回示例

`XML` 格式

```
<TotalCount>4</TotalCount>
<TotalPage>1</TotalPage>
<PageSize>10</PageSize>
<RequestId>F33B93C0-56E6-43BD-B30F-E1FF1D4EF1EE</RequestId>
<Packages>
    <Package>
        <ModifyTime>1600156643000</ModifyTime>
        <OriginName>aliyun-java-sdk-foas-2.9.0.jar</OriginName>
        <Type>JAR</Type>
        <Description>adas</Description>
        <ProjectName>project1</ProjectName>
        <PackageName>aliyun-java-sdk-foas-2.9.0.jar</PackageName>
        <CreateTime>1600156643000</CreateTime>
        <OssPath>79zec7svps7hca0xnogbqpvu/aliyun-java-sdk-foas-2.9.0.jar/l1jsjfzlps4z4pk5rh
utb91k.bayes</OssPath>
        <Creator>1709064687573327</Creator>
        <Modifier>1709064687573327</Modifier>
        <Md5>e0899e291f0524c6e6572c126599f73b</Md5>
```

```
    </Package>
    <Package>
        <ModifyTime>1596436948000</ModifyTime>
        <OriginName>blink-udx-2.x-1.0-snapshot.jar</OriginName>
        <Type>JAR</Type>
        <Description>123</Description>
        <ProjectName>bayes_team</ProjectName>
        <PackageName>blink-udx-2.x-1.0-snapshot.jar</PackageName>
        <CreateTime>1596436947000</CreateTime>
        <OssPath>79zec7svps7hca0xnogbqpvu/blink-udx-2.x-1.0-snapshot.jar/dr6r901verct65f612
ca4t0c.bayes</OssPath>
        <Creator>1709064687573327</Creator>
        <Modifier>1709064687573327</Modifier>
        <Md5>70bf694b7c737ee608f1aaf11b15816a</Md5>
    </Package>
    <Package>
        <ModifyTime>1551161104000</ModifyTime>
        <OriginName>blink_test_datastream.jar</OriginName>
        <Type>JAR</Type>
        <Description>1</Description>
        <ProjectName>bayes_team</ProjectName>
        <PackageName>blink_test_datastream.jar</PackageName>
        <CreateTime>1551161103000</CreateTime>
        <OssPath>79zec7svps7hca0xnogbqpvu/blink_test_datastream.jar/mg5174rvvsv472xrkvv23wn
3.bayes</OssPath>
        <Creator>1709064687573327</Creator>
        <Modifier>1709064687573327</Modifier>
        <Md5>7fd9782e4aad6e4b6a9aed8521774d40</Md5>
    </Package>
    <Package>
        <ModifyTime>1546094734000</ModifyTime>
        <OriginName>sql_udx-3.0</OriginName>
        <Type>JAR</Type>
        <Description>1</Description>
        <ProjectName>bayes_team</ProjectName>
        <PackageName>sql_udx-3.0</PackageName>
        <CreateTime>1546094733000</CreateTime>
        <OssPath>79zec7svps7hca0xnogbqpvu/sql_udx-3.0/6za6dxizvsscdjpupnbp0cfq.bayes</OssPa
th>
        <Creator>1709064687573327</Creator>
        <Modifier>1709064687573327</Modifier>
        <Md5>d8e19321c1b33af2e52643ea169d4f7d</Md5>
    </Package>
</Packages>
<PageIndex>1</PageIndex>
```

`JSON` 格式

```
{
 "TotalCount": 4,
 "TotalPage": 1,
 "PageSize": 10,
 "RequestId": "F33B93C0-56E6-43BD-B30F-E1FF1D4EF1EE",
 "Packages": {
```

```
  "Package": [
    {
     "ModifyTime": 1600156643000,
     "OriginName": "aliyun-java-sdk-foas-2.9.0.jar",
     "Type": "JAR",
     "Description": "adas",
     "ProjectName": "project1",
     "PackageName": "aliyun-java-sdk-foas-2.9.0.jar",
     "CreateTime": 1600156643000,
     "OssPath": "79zec7svps7hca0xnogbqpvu/aliyun-java-sdk-foas-2.9.0.jar/l1jsjfzlps4z4pk5rhu
tb91k.bayes",
     "Creator": "1709064687573327",
     "Modifier": "1709064687573327",
     "Md5": "e0899e291f0524c6e6572c126599f73b"
    },
    {
     "ModifyTime": 1596436948000,
     "OriginName": "blink-udx-2.x-1.0-snapshot.jar",
     "Type": "JAR",
     "Description": "123",
     "ProjectName": "bayes_team",
     "PackageName": "blink-udx-2.x-1.0-snapshot.jar",
     "CreateTime": 1596436947000,
     "OssPath": "79zec7svps7hca0xnogbqpvu/blink-udx-2.x-1.0-snapshot.jar/dr6r901verct65f612c
a4t0c.bayes",
     "Creator": "1709064687573327",
     "Modifier": "1709064687573327",
     "Md5": "70bf694b7c737ee608f1aaf11b15816a"
    },
    {
     "ModifyTime": 1551161104000,
     "OriginName": "blink_test_datastream.jar",
     "Type": "JAR",
     "Description": "1",
     "ProjectName": "bayes_team",
     "PackageName": "blink_test_datastream.jar",
     "CreateTime": 1551161103000,
     "OssPath": "79zec7svps7hca0xnogbqpvu/blink_test_datastream.jar/mg5174rvvsv472xrkvv23wn3
.bayes",
     "Creator": "1709064687573327",
     "Modifier": "1709064687573327",
     "Md5": "7fd9782e4aad6e4b6a9aed8521774d40"
    },
    {
     "ModifyTime": 1546094734000,
     "OriginName": "sql_udx-3.0",
     "Type": "JAR",
     "Description": "1",
     "ProjectName": "bayes_team",
     "PackageName": "sql_udx-3.0",
     "CreateTime": 1546094733000,
     "OssPath": "79zec7svps7hca0xnogbqpvu/sql_udx-3.0/6za6dxizvsscdjpupnbp0cfq.bayes",
     "Creator": "1709064687573327",
     "Modifier": "1709064687573327",
```

```
    "Md5": "d8e19321c1b33af2e52643ea169d4f7d"
   }
  ]
 },
 "PageIndex": 1
}
```

8.6. UpdatePackage

您可以通过UpdatePackage OpenAPI更新开发控制台上保存在指定项目下的Package。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
PUT /api/v2/projects/[projectName]/packages/[packageName] HTTPS
```

请求参数

名称	类型	是否必选	示例值	描述
packageName	String	是	package1.jar	Package名称
projectName	String	是	project1	项目名称
RegionId	String	否	cn-hangzhou	区域ID **说明** 公共云用户请忽略此参数。
originName	String	否	package2.jar	Package别名
description	String	否	test	Package备注描述
md5	String	否	3F7468C153E529B141C326332DF15D05	Package的Md5值

名称	类型	是否必选	示例值	描述
ossEndpoint	String	否	oss-cn-hangzhou-internal.aliyuncs.com	OSS接入点
ossBucket	String	否	blinktest2.oss-cn-hangzhou-internal.aliyuncs.com	OSS Bucket
ossOwner	String	否	owner1	OSS所有者
ossPath	String	否	path1/path2/a.jar	OSS路径
tag	String	否	{"function":"group by time"}	Package的标记

返回数据

名称	类型	示例值	描述
RequestId	String	77D5CB98-5E9F-42BB-8AA9-001125B48E28	请求ID

示例

请求示例

```
/api/v2/projects/project1/packages/package1.jar
```

正常返回示例

`XML` 格式

```
<RequestId>77D5CB98-5E9F-42BB-8AA9-001125B48E28</RequestId>
```

`JSON` 格式

```
{
 "RequestId": "77D5CB98-5E9F-42BB-8AA9-001125B48E28"
}
```

9.Queue

9.1. CreateQueue

您可以使用CreateQueue将集群资源分配到创建的Queue中，并绑定Queue中的项目。Queue创建成功后即开始消耗资源。

> **说明** CreateQueue仅限独享模式使用。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
POST /api/v2/clusters/[clusterId]/queue HTTP/1.1
```

请求参数

名称	类型	位置	是否必选	示例值	描述
clusterId	String	Path	是	cmy99ugusuco66x9qc6k****	集群ID。 **说明** 您可以使用listcluster获取集群ID信息。
maxMemMB	Integer	Body	是	16	Queue的最大内存，单位为MB。
maxVcore	Integer	Body	是	4	Queue的最大CPU个数。 **说明** 100Vcore = 1CU = 4G MEM。
queueName	String	Body	是	root.default	Queue的名称。
RegionId	String	Header	否	cn-shanghai	地域ID。

名称	类型	位置	是否必选	示例值	描述
gpu	Integer	Body	否	1	GPU个数。

返回数据

名称	类型	示例值	描述
RequestId	String	9F1CAD8D-E80E-45AF-82D7-8314FE17A34A	请求ID

示例

请求示例

```
/api/v2/clusters/cmy99ugusuco66x9qc6k****/queue
```

正常返回示例

`XML` 格式

```
<RequestId>9F1CAD8D-E80E-45AF-82D7-8314FE17A34A</RequestId>
```

`JSON` 格式

```
{
    "RequestId":"9F1CAD8D-E80E-45AF-82D7-8314FE17A34A"
}
```

9.2. DeleteQueue

您可以通过DeleteQueue OpenAPI删除已存在的Queue。如果需要删除的Queue已被项目绑定，则需要先解绑项目，再进行删除。

> **说明** DeleteQueue仅适用于独享模式。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
DELETE /api/v2/clusters/[clusterId]/queue HTTPS
```

请求参数

名称	类型	是否必选	示例值	描述
clusterId	String	是	d6wxwo5tnrmuamx2ly3m****	集群ID **说明** 您可以使用listcluster获取集群ID信息。
queueName	String	是	root.default	Queue名称 **说明** 您可以使用GetClusterQueueInfo获取Queue名称信息。
RegionId	String	是	cn-hangzhou	区域ID **说明** 公共云用户请忽略此参数。

返回数据

名称	类型	示例值	描述
RequestId	String	C5C6746D-2FFC-4D5D-A6A8-FA8DF7585DC2	请求ID

示例

请求示例

```
/api/v2/clusters/d6wxwo5tnrmuamx2ly3m****/queue
```

正常返回示例

`XML` 格式

```
<requestId>C5C6746D-2FFC-4D5D-A6A8-FA8DF7585DC2</requestId>
```

`JSON` 格式

```
{"requestId":"C5C6746D-2FFC-4D5D-A6A8-FA8DF7585DC2"}
```

9.3. BindQueue

您可以通过BindQueue OpenAPI将Project绑定在已经存在但没有被其他Project绑定的Queue上，绑定成功后，系统才会给您的Project分配资源，您才能在该项目上创建和运行作业。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
POST /api/v2/projects/[projectName]/queue HTTPS
```

请求参数

名称	类型	是否必选	示例值	描述
clusterId	String	是	d6wxwo5tnrmuamx2ly3m****	集群ID 说明 您可以使用listcluster获取集群ID信息。
projectName	String	是	project1	项目名称
queueName	String	是	queue1	要绑定的Queue名称 说明 您可以使用GetClusterQueueInfo获取Queue名称信息。
RegionId	String	否	cn-hangzhou	区域ID 说明 公共云用户请忽略此参数。

返回数据

名称	类型	示例值	描述

名称	类型	示例值	描述
RequestId	String	C5C6746D-2FFC-4D5D-A6A8-FA8DF7585DC2	请求ID

示例

请求示例

```
/api/v2/projects/project1/queue
{\"clusterId\":\"d6wxwo5tnrmuamx2ly3m****\",\"queueName\":\"queue1\"}
```

正常返回示例

`XML` 格式

```
<requestId>C5C6746D-2FFC-4D5D-A6A8-FA8DF7585DC2</requestId>
```

`JSON` 格式

```
{"requestId":"C5C6746D-2FFC-4D5D-A6A8-FA8DF7585DC2"}
```

9.4. UnbindQueue

您可以通过UnbindQueue OpenAPI将Queue中已经绑定的项目进行解绑。执行UnbindQueue操作前，需要将项目上的作业全部下线。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
DELETE /api/v2/projects/[projectName]/queue HTTPS
```

请求参数

名称	类型	是否必选	示例值	描述

名称	类型	是否必选	示例值	描述
clusterId	String	是	cmy99ugusuco66x9qc6k****	集群ID **说明** 您可以使用listcluster获取集群ID信息。
projectName	String	是	project1	项目名称
queueName	String	是	queue1	要绑定的Queue名称 **说明** 您可以使用GetClusterQueueInfo获取Queue名称信息。
RegionId	String	否	cn-hangzhou	区域ID **说明** 公共云用户请忽略此参数。

返回数据

名称	类型	示例值	描述
RequestId	String	C5C6746D-2FFC-4D5D-A6A8-FA8DF7585DC2	请求ID

示例

请求示例

```
/api/v2/projects/project1/queue
```

正常返回示例

`XML` 格式

```
<requestId>C5C6746D-2FFC-4D5D-A6A8-FA8DF7585DC2</requestId>
```

`JSON` 格式

```
{"requestId":"C5C6746D-2FFC-4D5D-A6A8-FA8DF7585DC2"}
```

9.5. GetClusterQueueInfo

您可以通过GetClusterQueueInfo查询集群上已存在的Queue的信息。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
GET /api/v2/clusters/[clusterId]/queueinfo HTTP/1.1
```

请求参数

名称	类型	位置	是否必选	示例值	描述
clusterId	String	Path	是	h6272cj4etgqe7oets4s****	集群ID。 **说明** 您可以使用listcluster获取集群ID信息。
RegionId	String	Header	是	cn-hangzhou	区域ID。

返回数据

名称	类型	示例值	描述
Queues	Array of Queue		返回Queues详情。
Queue			
ClusterId	String	h6272cj4etgqe7oets4s****	集群ID。 **说明** 您可以使用listcluster获取集群ID信息。

名称	类型	示例值	描述
ExternalInfo	String	{\"minGpu\":0,\"maxGpu\":0,\"name\":\"root.cloudpay_dev\",\"minMem\":24535,\"usedVCore\":50,\"usedGpu\":0,\"usedMem\":2816,\"minVCore\":599,\"maxVCore\":600,\"maxMem\":24576}	未转换为INTEGER时的原始信息。 **说明** • 仅在 `aliyun-java-sdk-foas 2.7.0` 及以上版本返回ExternalInfo。 • Integer类型返回-9999时，表示该数值超过Integer能表示的大小，请从ExternalInfo中取值。
MaxGpu	Integer	0	最大GPU。
MaxMem	Integer	24576	内存最大值（MB）。
MaxVCore	Integer	599	最大Vcore数。
MinGpu	Integer	0	最小GPU。
MinMem	Integer	24535	最小内存数（MB）。
MinVCore	Integer	599	最小Vcore数。
QueueName	String	root.cloudpay_dev	队列名称。
UsedGpu	Integer	0	已使用GPU。
UsedMem	Integer	2816	已使用内存（MB）。
UsedVCore	Integer	50	已使用Vcore。
RequestId	String	DACD3B42-5977-4293-B5A8-197A33A954BB	请求ID。

示例

请求示例

```
/api/v2/clusters/h6272cj4etgqe7oets4s****/queueinfo
```

正常返回示例

`XML` 格式

```
<RequestId>DACD3B42-5977-4293-B5A8-197A33A954BB</RequestId>
<Queues>
    <Queue>
        <MaxMem>24576</MaxMem>
        <UsedGpu>0</UsedGpu>
        <ExternalInfo>{"minGpu":0,"maxGpu":0,"name":"root.cloudpay_dev","minMem":24535,"use
dVCore":50,"usedGpu":0,"usedMem":2816,"minVCore":599,"maxVCore":600,"maxMem":24576}</Extern
alInfo>
        <UsedMem>2816</UsedMem>
        <ClusterId>h6272cj4etgqe7oets4s****</ClusterId>
        <MinGpu>0</MinGpu>
        <MaxVCore>600</MaxVCore>
        <MinVCore>599</MinVCore>
        <MaxGpu>0</MaxGpu>
        <MinMem>24535</MinMem>
        <QueueName>root.cloudpay_dev</QueueName>
        <UsedVCore>50</UsedVCore>
    </Queue>
</Queues>
```

`JSON` 格式

```
{
 "RequestId": "DACD3B42-5977-4293-B5A8-197A33A954BB",
 "Queues": {
  "Queue": [
   {
    "MaxMem": 24576,
    "UsedGpu": 0,
    "ExternalInfo": "{\"minGpu\":0,\"maxGpu\":0,\"name\":\"root.cloudpay_dev\",\"minMem\":2
4535,\"usedVCore\":50,\"usedGpu\":0,\"usedMem\":2816,\"minVCore\":599,\"maxVCore\":600,\"ma
xMem\":24576}",
    "UsedMem": 2816,
    "ClusterId": "h6272cj4etgqe7oets4s****",
    "MinGpu": 0,
    "MaxVCore": 600,
    "MinVCore": 599,
    "MaxGpu": 0,
    "MinMem": 24535,
    "QueueName": "root.cloudpay_dev",
    "UsedVCore": 50
   }
  ]
 }
}
```

9.6. ListProjectBindQueue

您可以通过ListProjectBindQueue OpenAPI查看指定项目对应的队列相关的信息。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
GET /api/v2/projects/[projectName]/queues HTTPS
```

请求参数

名称	类型	是否必选	示例值	描述
projectName	String	是	project1	项目名称
RegionId	String	否	cn-hangzhou	区域ID **说明** 公共云用户请忽略此参数。
clusterId	String	否	d6wxwo5tnrmuamx2ly3m****	集群ID **说明** 您可以使用listcluster获取集群ID信息。
queueName	String	否	root.project1	队列名称 **说明** 您可以使用GetClusterQueueInfo获取Queue名称信息。

返回数据

名称	类型	示例值	描述
Queues	Array of Queue		队列详情
Queue			

名称	类型	示例值	描述
ClusterId	String	d6wxwo5tnrmuamx2ly3m****	集群ID
QueueName	String	root.project1	队列名称
RequestId	String	36277E3E-99BA-43C7-8F91-ADCC8FE1EC96	请求ID

示例

请求示例

```
/api/v2/projects/project1/queues
```

正常返回示例

`XML` 格式

```
<RequestId>36277E3E-99BA-43C7-8F91-ADCC8FE1EC96</RequestId>
<Queues>
    <Queue>
        <ClusterId>d6wxwo5tnrmuamx2ly3m****</ClusterId>
        <QueueName>root.project1</QueueName>
    </Queue>
</Queues>
```

`JSON` 格式

```
{
 "RequestId": "36277E3E-99BA-43C7-8F91-ADCC8FE1EC96",
 "Queues": {
  "Queue": [
   {
    "ClusterId": "d6wxwo5tnrmuamx2ly3m****",
    "QueueName": "root.project1"
   }
  ]
 }
}
```

9.7. ListProjectBindQueueResource

您可以通过List Project Bind Queue Resource OpenAPI查询Project关联的队列详情。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
GET /api/v2/projects/[projectName]/queueresource HTTPS
```

请求参数

名称	类型	是否必选	示例值	描述
projectName	String	是	project1	项目名称
clusterId	String	否	d6wxwo5tnrmuamx2ly3m****	集群ID **说明** 您可以使用listcluster获取集群ID信息。
queueName	String	否	root.project1	队列名称 **说明** 您可以使用GetClusterQueueInfo获取Queue名称信息。
RegionId	String	否	cn-hangzhou	区域ID **说明** 公共云用户请忽略此参数。

返回数据

名称	类型	示例值	描述
Queues	Array of Queue		队列详情
Queue			
ClusterId	String	d6wxwo5tnrmuamx2ly3m****	集群ID
MaxGpu	Integer	0	GPU最大值

名称	类型	示例值	描述
MaxMem	Integer	24576	内存最大值（MB）
MaxVCore	Integer	600	最大Vcore数
MinGpu	Integer	0	最小GPU数
MinMem	Integer	24535	最小内存数（MB）
MinVCore	Integer	599	最小Vcore数
QueueName	String	root.project1	队列名称
UsedGpu	Integer	0	已使用GPU
UsedMem	Integer	10496	已使用内存（MB）
UsedVCore	Integer	250	已使用VCore
RequestId	String	2127A47D-CB91-44AE-8277-3FC645A761CE	请求ID

示例

请求示例

```
/api/v2/projects/project1/queueresource
```

正常返回示例

`XML` 格式

```
<RequestId>2127A47D-CB91-44AE-8277-3FC645A761CE</RequestId>
<Queues>
    <Queue>
        <MaxMem>24576</MaxMem>
        <UsedGpu>0</UsedGpu>
        <UsedMem>10496</UsedMem>
        <ClusterId>d6wxwo5tnrmuamx2ly3m****</ClusterId>
        <MinGpu>0</MinGpu>
        <MaxVCore>600</MaxVCore>
        <MinVCore>599</MinVCore>
        <MaxGpu>0</MaxGpu>
        <MinMem>24535</MinMem>
        <QueueName>root.project1</QueueName>
        <UsedVCore>250</UsedVCore>
    </Queue>
</Queues>
```

`JSON` 格式

```
{
 "RequestId": "2127A47D-CB91-44AE-8277-3FC645A761CE",
 "Queues": {
  "Queue": [
   {
    "MaxMem": 24576,
    "UsedGpu": 0,
    "UsedMem": 10496,
    "ClusterId": "d6wxwo5tnrmuamx2ly3m****",
    "MinGpu": 0,
    "MaxVCore": 600,
    "MinVCore": 599,
    "MaxGpu": 0,
    "MinMem": 24535,
    "QueueName": "root.project1",
    "UsedVCore": 250
   }
  ]
 }
}
```

9.8. UpdateQueue

您可以通过UpdateQueue修改指定Queue的某些参数，例如maxVcore、maxMemMB等。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
PUT /api/v2/clusters/[clusterId]/queue HTTP/1.1
```

请求参数

名称	类型	位置	是否必选	示例值	描述
clusterId	String	Path	是	h6272cj4etgqe7oe****	集群ID。 **说明** 您可以使用listcluster获取集群ID信息。
maxMemMB	Integer	Body	是	16	Queue的最大内存，单位为MB。
maxVcore	Integer	Body	是	4	Queue的最大CPU个数。 **说明** 100Vcore=1core。
queueName	String	Body	是	root.cloudpay_dev	Queue的名称。
RegionId	String	Header	否	cn-hangzhou	地域。
gpu	Integer	Body	否	1	GPU个数。

返回数据

名称	类型	示例值	描述
RequestId	String	9F1CAD8D-E80E-45AF-82D7-8314FE17A34A	请求ID。

示例

请求示例

```
/api/v2/clusters/h6272cj4etgqe7oe****/queue
```

正常返回示例

`XML` 格式

```
<RequestId>9F1CAD8D-E80E-45AF-82D7-8314FE17A34A</RequestId>
```

`JSON` 格式

```
{
    "RequestId":"9F1CAD8D-E80E-45AF-82D7-8314FE17A34A"
}
```

10.Cluster

10.1. CreateCluster

您可以在提交实时计算独享模式购买订单后，通过CreateCluster OpenAPI创建集群。

> **说明** 仅独享模式支持CreateCluster。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
POST /api/v2/clusters HTTP/1.1
```

请求参数

名称	类型	位置	是否必选	示例值	描述
description	String	Body	是	cluster1	集群描述。
displayName	String	Body	是	cluster_name	集群名称。
orderId	String	Body	是	blinkonecs_1234	购买订单生成的实例ID。
userOssBucket	String	Body	是	12345	您OSS中Bucket名称。
userVpcId	String	Body	是	vpc-abcde	您的集群所在VPC名称。 **说明** 实时计算和连通的上下游数据存储必须部署于同一个VPC，否则会造成数据无法连通的问题。
userVSwitch	String	Body	是	vsw-abcde	交换机名称。

名称	类型	位置	是否必选	示例值	描述
zoneId	String	Body	是	cn-shanghai-f	可用区。
RegionId	String	Header	否	cn-hangzhou	区域。

返回数据

名称	类型	示例值	描述
ClusterId	String	pgr1rnrdlry05mhsarcj****	集群ID。 说明 您可以使用listcluster获取集群ID信息。
RequestId	String	9F1CAD8D-E80E-45AF-82D7-8314FE17A34A	请求ID。

示例

请求示例

```
{\"zoneId\":\"cn-shanghai-f\",\"displayName\":\"cluster_name\",\"description\":\"cluster1\",\"userOssBucket\":\"12345\",\"userVpcId\":\"vpc-abcde\",\"userVSwitch\":\"vsw-abcde\",\"orderId\":\"blinkonecs_1234\"}
```

正常返回示例

`XML` 格式

```
<RequestId>9F1CAD8D-E80E-45AF-82D7-8314FE17A34A</RequestId>
<ClusterId>pgr1rnrdlry05mhsarcj****</ClusterId>
```

`JSON` 格式

```
{
    "RequestId":"9F1CAD8D-E80E-45AF-82D7-8314FE17A34A",
    "ClusterId":"pgr1rnrdlry05mhsarcj****"
}
```

10.2. CreateCellClusterOrder

您可以通过CreateCellClusterOrder创建一个实时计算的实例，仅限独享集群。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
POST /api/v2/realtime-compute/cell/buy HTTP/1.1
```

请求参数

名称	类型	位置	是否必选	示例值	描述
masterNum	Integer	Body	是	3	Master机器的数量，仅支持1台或者3台。 **说明** • 您需要3台master才能有效保障服务SLA。 • 选择该参数后不支持修改。
masterSpec	String	Body	是	Ecs_4c16g	Master机器型号。
payModel	String	Body	是	pre	实例类型： **说明** • pre（包年包月）。 • post（按量计费）。
region	String	Body	是	cn-hangzhou	地域。
slaveNum	Integer	Body	是	3	Slave机器数量，至少2台。
slaveSpec	String	Body	是	Ecs_8c32g	Slave机器型号。
period	Integer	Body	否	8	仅pre（包年包月）填写，表示购买时间，最多可以购买12个月，单位是月。

名称	类型	位置	是否必选	示例值	描述
RegionId	String	Header	否	cn-shanghai	地域。

返回数据

名称	类型	示例值	描述
OrderId	String	blinkonecs_post-cn-6ja214s9y06	订单号。
RequestId	String	6676DF01-0533-4CB0-8EEB-74BDB1848A3B	请求ID。

示例

请求示例

```
/api/v2/realtime-compute/cell/buy
{\"region\":\"cn-shanghai\",\"masterSpec\":\"Ecs_4c16g\",\"masterNum\":3,\"slaveSpec\":\"Ecs_8c32g\",\"slaveNum\":3,\"payModel\":\"post\"}
```

正常返回示例

`XML` 格式

```
<RequestId>6676DF01-0533-4CB0-8EEB-74BDB1848A3B</RequestId>
<OrderId>blinkonecs_post-cn-6ja214s9y06</OrderId>
```

`JSON` 格式

```
{
    "RequestId": "6676DF01-0533-4CB0-8EEB-74BDB1848A3B",
    "OrderId": "blinkonecs_post-cn-6ja214s9y06"
}
```

10.3. DestroyCluster

您可以通过DestroyCluster OpenAPI注销集群，集群注销后不再产生费用。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
DELETE /api/v2/clusters/[clusterId] HTTPS
```

请求参数

名称	类型	是否必选	示例值	描述
clusterId	String	是	cmy99ugusuco66x9qc6k****	集群ID **说明** 您可以使用listcluster获取集群ID信息。
RegionId	String	是	cn-hangzhou	区域ID **说明** 公共云用户请忽略此参数。

返回数据

名称	类型	示例值	描述
RequestId	String	C5C6746D-2FFC-4D5D-A6A8-FA8DF7585DC2	请求ID

示例

请求示例

```
/api/v2/clusters/cmy99ugusuco66x9qc6k****
```

正常返回示例

`XML` 格式

```
<requestId>C5C6746D-2FFC-4D5D-A6A8-FA8DF7585DC2</requestId>
```

`JSON` 格式

```
{"requestId":"C5C6746D-2FFC-4D5D-A6A8-FA8DF7585DC2"}
```

10.4. ExpandCluster

您可以通过ExpandCluster OpenAPI对已有集群进行Slave机型的扩容。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
PUT /api/v2/clusters/[clusterId]/expand HTTPS
```

请求参数

名称	类型	是否必选	示例值	描述
clusterId	String	是	cmy99ugusuco66x9qc6k****	集群ID **说明** 您可以使用listcluster获取集群ID信息。
RegionId	String	是	cn-hangzhou	区域ID **说明** 公共云用户请忽略此参数。
model	String	否	Ecs_4c16g	机器型号 **说明** 扩容机型必须和已有Slave机型一致。
count	Integer	否	5	扩容后的机器台数
userVSwitch	String	否	vsw-abcd	交换机名称 **说明** 扩容提示vSwitch不足时，需要填写该参数。

返回数据

名称	类型	示例值	描述
RequestId	String	C5C6746D-2FFC-4D5D-A6A8-FA8DF7585DC2	请求ID

示例

请求示例

```
/api/v2/clusters/cmy99ugusuco66x9qc6k****/expand
```

正常返回示例

`XML` 格式

```
<requestId>C5C6746D-2FFC-4D5D-A6A8-FA8DF7585DC2</requestId>
```

`JSON` 格式

```
{"requestId":"C5C6746D-2FFC-4D5D-A6A8-FA8DF7585DC2"}
```

10.5. ShrinkCluster

您可以通过ShrinkCluster来减少集群Slave实例台数。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
PUT /api/v2/clusters/[clusterId]/shrink HTTP/1.1
```

请求参数

名称	类型	位置	是否必选	示例值	描述
clusterId	String	Path	是	cmy99ugusuco66x9qc6k****	集群ID。 **说明** 您可以使用listcluster获取集群ID信息。

名称	类型	位置	是否必选	示例值	描述
instanceIds	String	Body	是	403936	集群Slave实例ID。指定多个实例需用逗号（,）分割。 您可以通过getClusterDetails接口查看已创建的集群的配置信息。 如果同时配置instanceIds和modelTargetCount参数时，以instanceIds为准。
RegionId	String	Header	否	cn-hangzhou	区域ID。
modelTargetCount	String	Body	否	{Ecs_4c16g:2}	指定缩容实例的型号和台数。 **说明** 缩容过程中随机去除Slave实例。

返回数据

名称	类型	示例值	描述
RequestId	String	5A0AF0E2-A715-40FD-8A2B-1EDBF4213489	请求ID。

示例

请求示例

```
/api/v2/clusters/cmy99ugusuco66x9qc6k****/shrink
{\"instanceIds\":\"403936\"}
```

正常返回示例

`XML` 格式

```
<RequestId>5A0AF0E2-A715-40FD-8A2B-1EDBF4213489</RequestId>
```

`JSON` 格式

```
{
    "RequestId": "5A0AF0E2-A715-40FD-8A2B-1EDBF4213489"
}
```

10.6. ListCluster

您可以通过List Cluster OpenAPI查询集群信息。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
GET /api/v2/clusters HTTP/1.1
```

请求参数

名称	类型	位置	是否必选	示例值	描述
clusterId	String	Query	否	rulpqd442sgxbtw9736r****	集群ID。
displayName	String	Query	否	demo	集群名称。
state	String	Query	否	RUNNING	集群状态： • STARTING： 集群创建中。 • EXPANDING：集群扩容中（增加Slave机型台数）。 • UPGRADING： 集群变配中（提高或者降低Master）。 • DESTROYING： 集群注销中。 • DESTROYED： 集群已销毁。 • REDUCING：集群缩容中（减少Slave机型台数）。 • MAINTAINING：集群维护中。
region	String	Query	否	cn-beijing	区域。
pageSize	Integer	Query	否	10	每页显示的集群数。
pageIndex	Integer	Query	否	1	指定查询的页码。
RegionId	String	Header	否	cn-hangzhou	区域ID。

返回数据

名称	类型	示例值	描述
Clusters	Array of Cluster		集群详情。
Cluster			
ClusterId	String	rulpqd442sgxbtw9736r****	集群ID。
Description	String	demo	集群备注。
DisplayName	String	demo	集群名称。
GmtCreate	Long	1605609438000	集群VPC创建时间。
GmtModified	Long	1605610377000	集群VPC修改时间。
Operator	String	27395716024545****	集群最后操作者UID。
OwnerId	String	107992689699****	集群拥有者UID。
RegionId	String	cn-beijing	集群所属地区。
State	String	RUNNING	集群状态： • STARTING： 集群创建中。 • EXPANDING：集群扩容中（增加Slave机型台数）。 • UPGRADING： 集群变配中（提高或者降低Master）。 • DESTROYING： 集群注销中。 • DESTROYED： 集群已销毁。 • REDUCING：集群缩容中（减少Slave机型台数）。 • MAINTAINING：集群维护中。
ZoneId	String	cn-beijing-e	区域ID。
PageIndex	Integer	1	指定查询的页码。
PageSize	Integer	10	每页显示的集群数。

名称	类型	示例值	描述
RequestId	String	10C4F123-351D-4D2B-94A4-D6FF456AEA6E	请求ID。
TotalCount	Long	3	总集群数。
TotalPage	Integer	10	总页数。

示例

请求示例

```
http(s)://[Endpoint]/?RegionId=cn-hangzhou&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<TotalCount>3</TotalCount>
<TotalPage>1</TotalPage>
<PageSize>10</PageSize>
<RequestId>10C4F123-351D-4D2B-94A4-D6FF456AEA6E</RequestId>
<Clusters>
    <Cluster>
        <Operator>27395716024545****</Operator>
        <GmtCreate>1605609438000</GmtCreate>
        <Description>demo</Description>
        <ZoneId>cn-beijing-e</ZoneId>
        <OwnerId>107992689699****</OwnerId>
        <ClusterId>rulpqd442sgxbtw9736r****</ClusterId>
        <State>RUNNING</State>
        <DisplayName>demo</DisplayName>
        <GmtModified>1605610377000</GmtModified>
        <RegionId>cn-beijing</RegionId>
    </Cluster>
    <Cluster>
        <Operator>23996037709276****</Operator>
        <GmtCreate>1597654146000</GmtCreate>
        <Description>test</Description>
        <ZoneId>cn-hangzhou-f</ZoneId>
        <OwnerId>107992689699****</OwnerId>
        <ClusterId>rcmp9x37ztfb63g1x7lt****</ClusterId>
        <State>RUNNING</State>
        <DisplayName>niuna</DisplayName>
        <GmtModified>1601051659000</GmtModified>
        <RegionId>cn-hangzhou</RegionId>
    </Cluster>
</Clusters>
<PageIndex>1</PageIndex>
```

JSON 格式


```
{
 "TotalCount": 3,
 "TotalPage": 1,
 "PageSize": 10,
 "RequestId": "10C4F123-351D-4D2B-94A4-D6FF456AEA6E",
 "Clusters": {
  "Cluster": [
   {
    "Operator": "273957160245454****",
    "GmtCreate": 1605609438000,
    "Description": "demo",
    "ZoneId": "cn-beijing-e",
    "OwnerId": "107992689699****",
    "ClusterId": "rulpqd442sgxbtw9736r****",
    "State": "RUNNING",
    "DisplayName": "demo",
    "GmtModified": 1605610377000,
    "RegionId": "cn-beijing"
   },
   {
    "Operator": "239960377092764****",
    "GmtCreate": 1597654146000,
    "Description": "test",
    "ZoneId": "cn-hangzhou-f",
    "OwnerId": "107992689699****",
    "ClusterId": "rcmp9x37ztfb63g1x7lt****",
    "State": "RUNNING",
    "DisplayName": "niuna",
    "GmtModified": 1601051659000,
    "RegionId": "cn-hangzhou"
   }
  ]
 },
 "PageIndex": 1
}
```


10.7. GetClusterDetails

您可以通过GetClusterDetails查询已创建的集群的配置信息。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
GET /api/v2/clusters/[clusterId]/details HTTP/1.1
```

请求参数

名称	类型	位置	是否必选	示例值	描述
clusterId	String	Path	是	cmy99ugusuco66x9qc6k****	集群ID。 说明 您可以使用listcluster获取集群ID信息。
RegionId	String	Header	是	cn-hangzhou	区域ID。

返回数据

名称	类型	示例值	描述
Details	Struct		集群配置详情。
ClusterId	String	7cmedvslzs8q9n7pkfx5****	集群ID。
Description	String	test	集群描述。
DisplayName	String	test	集群名称。
GmtCreate	Long	1612693397000	集群VPC创建时间。
GmtModified	Long	1612693397000	集群VPC修改时间。
InstanceInfos	String	{k:v}	集群Master机型和Slave机型的相关信息，包括实例ID，ENI，IP和机器型号等。
IsMixDeploy	Boolean	false	是否为混部集群（存在多种型号Slave机型）。
Operator	String	137677185440****	集群最后操作人UID。
OwnerId	String	137677185440****	集群拥有者UID。

名称	类型	示例值	描述
RegionId	String	cn-shanghai	区域ID。
State	String	STARTING	集群状态： • 集群创建中： STARTING。 • 集群扩容中（增加Slave机型台数）：EXPANDING。 • 集群变配中（提高或者降低Master型号）：UPGRADING。 • 集群注销中： DESTROYING。 • 集群已销毁： DESTROYED。 • 集群缩容中（减少Slave机型台数）：REDUCING。 • 集群维护中：MAINTAINING。
StorageType	String	HDFS	存储类型。
UserOssInfo	String	{\"endpoint\":\"oss-cn-shanghai.aliyuncs.com\",\"vpcEndpoint\":\"oss-cn-shanghai-internal.aliyuncs.com\",\"oxsEndpoint\":\"oss-cn-shanghai-internal.aliyuncs.com\",\"accessId\":null,\"accessKey\":null,\"bucket\":\"flink-test-oss-beijing\"}	您的OSS信息。以JSO形式返回，包括Bucket，Endpoint等。
UserSGId	String	sg-abcd	安全组ID。
UserVSwitchList	String	[\"vsw-uf6v128q9khll8e9ucyhj\"]	您的VSW列表。
UserVpcId	String	vpc-uf6aq156h1gy06dzf****	VPC ID。
ZoneId	String	cn-shanghai-f	可用地域ID。

名称	类型	示例值	描述
RequestId	String	A35C7747-AE0B-4688-AD84-82CAAD512FD3	请求ID。

示例

请求示例

```
/api/v2/clusters/cmy99ugusuco66x9qc6k****/details
```

正常返回示例

`XML` 格式

```
<Details>
    <Operator>137677185440****</Operator>
    <UserOssInfo>{"endpoint":"oss-cn-shanghai.aliyuncs.com","vpcEndpoint":"oss-cn-shanghai-internal.aliyuncs.com","oxsEndpoint":"oss-cn-shanghai-internal.aliyuncs.com","accessId":null,"accessKey":null,"bucket":"flink-test-oss-beijing"}</UserOssInfo>
    <Description>test</Description>
    <StorageType>HDFS</StorageType>
    <ZoneId>cn-shanghai-f</ZoneId>
    <UserVSwitchList>["vsw-uf6v128q9khll8e9ucyhj"]</UserVSwitchList>
    <ClusterId>7cmedvslzs8q9n7pkfx5****</ClusterId>
    <IsMixDeploy>false</IsMixDeploy>
    <GmtModified>1612693397000</GmtModified>
    <GmtCreate>1612693397000</GmtCreate>
    <OwnerId>137677185440****</OwnerId>
    <State>STARTING</State>
    <UserVpcId>vpc-uf6aq156h1gy06dzf****</UserVpcId>
    <DisplayName>test</DisplayName>
    <RegionId>cn-shanghai</RegionId>
</Details>
<RequestId>A35C7747-AE0B-4688-AD84-82CAAD512FD3</RequestId>
```

`JSON` 格式

```
{
 "Details": {
  "Operator": "137677185440****",
  "UserOssInfo": "{\"endpoint\":\"oss-cn-shanghai.aliyuncs.com\",\"vpcEndpoint\":\"oss-cn-s
hanghai-internal.aliyuncs.com\",\"oxsEndpoint\":\"oss-cn-shanghai-internal.aliyuncs.com\",\
"accessId\":null,\"accessKey\":null,\"bucket\":\"flink-test-oss-beijing\"}",
  "Description": "test",
  "StorageType": "HDFS",
  "ZoneId": "cn-shanghai-f",
  "UserVSwitchList": "[\"vsw-uf6v128q9khll8e9ucyhj\"]",
  "ClusterId": "7cmedvslzs8q9n7pkfx5****",
  "IsMixDeploy": false,
  "GmtModified": 1612693397000,
  "GmtCreate": 1612693397000,
  "OwnerId": "137677185440****",
  "State": "STARTING",
  "UserVpcId": "vpc-uf6aq156h1gy06dzf****",
  "DisplayName": "test",
  "RegionId": "cn-shanghai"
 },
 "RequestId": "A35C7747-AE0B-4688-AD84-82CAAD512FD3"
}
```


10.8. GetClusterEngineVersions

您可以通过GetClusterEngineVersions OpenAPI获取集群上可使用的引擎版本号。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
GET /api/v2/clusters/[clusterId]/engineversions HTTPS
```

请求参数

名称	类型	是否必选	示例值	描述
clusterId	String	是	cmy99ugusuco66x9qc6k****	集群ID **说明** 您可以使用listcluster获取集群ID信息。

名称	类型	是否必选	示例值	描述
RegionId	String	是	cn-hangzhou	区域ID **说明** 公共云用户请忽略此参数。

返回数据

名称	类型	示例值	描述
EngineVersions	List	current--blink-3.4.4	引擎版本
RequestId	String	7203295B-3726-4BEF-B3E1-9EE33BF2FECF	请求ID

示例

请求示例

```
/api/v2/clusters/cmy99ugusuco66x9qc6k****/engineversions
```

正常返回示例

`XML` 格式

```
<RequestId>7203295B-3726-4BEF-B3E1-9EE33BF2FECF</RequestId>
<EngineVersions>
    <EngineVersion>current--blink-3.4.4</EngineVersion>
</EngineVersions>
```

`JSON` 格式

```
{
 "RequestId": "7203295B-3726-4BEF-B3E1-9EE33BF2FECF",
 "EngineVersions": {
  "EngineVersion": [
   "current--blink-3.4.4"
  ]
 }
}
```

10.9. GetClusterResource

您可以通过GetClusterResource OpenAPI查看集群总资源和当前已使用的资源情况。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
GET /api/v2/clusters/[clusterId]/resource HTTPS
```

请求参数

名称	类型	是否必选	示例值	描述
clusterId	String	是	cmy99ugusuco66x9qc6k****	集群ID 说明 您可以使用listcluster获取集群ID信息。
RegionId	String	是	cn-hangzhou	区域ID 说明 公共云用户请忽略此参数。

返回数据

名称	类型	示例值	描述
RequestId	String	1F721BC4-6298-42BB-9289-794E7B0349B9	请求ID
Resource	Struct		资源详情
AllocatedMB	Long	10496	已使用的内存
AllocatedVirtualCores	Long	250	已使用的VCore数
AvailableMB	Long	14080	可使用的内存
AvailableVirtualCores	Long	350	可使用的VCore数

名称	类型	示例值	描述
TotalMB	Long	24576	集群总内存
TotalVirtualCores	Long	600	集群总VCore数

示例

请求示例

```
/api/v2/clusters/cmy99ugusuco66x9qc6k****/resource
```

正常返回示例

`XML` 格式

```
<RequestId>1F721BC4-6298-42BB-9289-794E7B0349B9</RequestId>
<Resource>
    <TotalVirtualCores>600</TotalVirtualCores>
    <AllocatedMB>10496</AllocatedMB>
    <AllocatedVirtualCores>250</AllocatedVirtualCores>
    <AvailableVirtualCores>350</AvailableVirtualCores>
    <TotalMB>24576</TotalMB>
    <AvailableMB>14080</AvailableMB>
</Resource>
```

`JSON` 格式

```
{
 "RequestId": "1F721BC4-6298-42BB-9289-794E7B0349B9",
 "Resource": {
  "TotalVirtualCores": 600,
  "AllocatedMB": 10496,
  "AllocatedVirtualCores": 250,
  "AvailableVirtualCores": 350,
  "TotalMB": 24576,
  "AvailableMB": 14080
 }
}
```

10.10. ModifyMasterSpec

您可以通过ModifyMasterSpec OpenAPI更改集群Master机型的规格，实现Master机型的升配和降配。

> **说明** ModifyMasterSpec仅支持在独享模式的按量付费模式下使用。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
PUT /api/v2/clusters/[clusterId]/specification HTTPS
```

请求参数

名称	类型	是否必选	示例值	描述
clusterId	String	是	cmy99ugusuco66x9qc6k****	集群ID 说明 您可以使用listcluster获取集群ID信息。
masterTargetModel	String	是	Ecs_4c16g	Master机型变配的目标机器型号
RegionId	String	否	cn-hangzhou	区域ID 说明 公共云用户请忽略此参数。

返回数据

名称	类型	示例值	描述
RequestId	String	C5C6746D-2FFC-4D5D-A6A8-FA8DF7585DC2	请求ID

示例

请求示例

```
/api/v2/clusters/cmy99ugusuco66x9qc6k****/specification
{\"masterTargetModel\":\"Ecs_4c16g\"}
```

正常返回示例

`XML` 格式

```
<requestId>C5C6746D-2FFC-4D5D-A6A8-FA8DF7585DC2</requestId>
```

JSON 格式

```
{"requestId":"C5C6746D-2FFC-4D5D-A6A8-FA8DF7585DC2"}
```

10.11. GetClusterMetrics

您可以通过GetClusterMetrics获取集群的CPU、存储或YARN的资源分配状况等信息。仅独享模式支持GetClusterMetrics接口。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

该接口使用公共请求头，无特殊请求头。请参见公共请求参数文档。

请求语法

```
POST /api/v2/clusters/[clusterId]/metrics HTTPS
```

请求参数

名称	类型	位置	是否必选	示例值	描述
clusterId	String	Path	是	yourClusterID	集群ID。<note>不是集群的名称。</note>
metricJson	String	Body	是	{ "end":1606291550061, "limit":"avg:sample:50", "queries":[{ "aggregator":"avg", "downsample":"avg", "granularity":"60s", "metric":"ClusterID.system.cpu.user" }], "start":1606291390000 }	查询Metric提交的JSON信息。
RegionId	String	Header	是	cn-hangzhou	地域ID

返回数据

名称	类型	示例值	描述
Metrics	Array of Metric		Metric信息。
Metric			
Dps	Map	k:v	时间点和对应的Metric值。
MetricName	String	delay	Metric名称。
Summary	Float	10.2	聚合后的Metric值。
Tags	Map	k:v	Metric标记。
RequestId	String	FD0FF8C0-779A-45EB-9674-FF3E127B10D2	请求ID。

示例

请求示例

```
POST /api/v2/clusters/[clusterId]/metrics HTTP/1.1
RegionId: cn-hangzhou
公共请求头
{
  "clusterId": "<yourClusterID>"
  "metricJson": "{metric: "clusterid.system.mem.memtotal", aggregator: "avg", downsample: "
avg"}"
}
```

正常返回示例

`XML` 格式

```
<RequestId>A017B343-33DC-4A97-BAC5-4B57872D5B61</RequestId>
<Metrics>
    <metricName>et2hunbu.yarn.resourcemanager.jvmmetrics.MemHeapUsed</metricName>
    <dps>
        <1574244620>3675.839599609375</1574244620>
        <1574244640>3817.712646484375</1574244640>
        <1574244660>3864.84130859375</1574244660>
        <1574244680>3973.963134765625</1574244680>
        <1574244700>4121.57958984375</1574244700>
        <1574244720>4170.6123046875</1574244720>
        <1574244740>4279.62109375</1574244740>
        <1574244760>4427.05615234375</1574244760>
        <1574244780>4472.11474609375</1574244780>
    </dps>
    <summary>6700.7467978583445</summary>
    <tags/>
</Metrics>
```

`JSON` 格式

```
{
    "RequestId":"A017B343-33DC-4A97-BAC5-4B57872D5B61",
    "Metrics":[
        {
            "metricName":"et2hunbu.yarn.resourcemanager.jvmmetrics.MemHeapUsed",
            "dps":{
                "1574244620":3675.839599609375,
                "1574244640":3817.712646484375,
                "1574244660":3864.84130859375,
                "1574244680":3973.963134765625,
                "1574244700":4121.57958984375,
                "1574244720":4170.6123046875,
                "1574244740":4279.62109375,
                "1574244760":4427.05615234375,
                "1574244780":4472.11474609375
            },
            "summary":6700.7467978583445,
            "tags":{
            }
        }
    ]
}
```

11.Project

11.1. CreateProject

您可以通过CreateProject OpenAPI在集群中创建项目。

> **说明** 不能通过子账号的AK信息以openAPI方式创建项目。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
POST /api/v2/projects HTTPS
```

请求参数

名称	类型	是否必选	示例值	描述
deployType	String	是	public	集群类型： • 独享集群：cell • 共享集群：public
name	String	否	project1	项目名称
orderId	String	否	stream-abcd	共享模式实例ID **说明** 仅共享模式需要填写。
clusterId	String	否	cmy99ugusuco66x9qc6k****	集群ID
managerIds	String	否	1234567	主账号ID
description	String	否	测试项目	项目描述

返回数据

名称	类型	示例值	描述
RequestId	String	B661F03F-0F9D-4EFF-A40F-4ED1519A549A	请求ID

示例

请求示例

```
/api/v2/projects
{\"name\":\"project1\",\"orderId\":\"stream-abcd\",\"clusterId\":\"cmy99ugusuco66x9qc6k****\",\"managerIds\":\"1234567\",\"description\":\"测试项目\",\"deployType\":\"cell\"}
```

正常返回示例

`XML` 格式

```
<requestId>B661F03F-0F9D-4EFF-A40F-4ED1519A549A</requestId>
```

`JSON` 格式

```
{"requestId":"B661F03F-0F9D-4EFF-A40F-4ED1519A549A"}
```

11.2. DeleteProject

您可以通过DeleteProject OpenAPI删除集群中的项目。如果需要删除的项目已经和Queue绑定，也会将项目和Queue解绑（不会删除Queue）。为了防止资源浪费，请及时将Queue绑定新的项目。

> **说明** DeleteProject仅支持独享模式。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
DELETE /api/v2/projects/[projectName] HTTPS
```

请求参数

名称	类型	是否必选	示例值	描述

名称	类型	是否必选	示例值	描述
projectName	String	是	project1	项目名称
RegionId	String	是	cn-hangzhou	区域ID **说明** 公共云用户请忽略此参数。

返回数据

名称	类型	示例值	描述
RequestId	String	3F4ADBCF-B116-4680-80B3-E9A139841A0D	请求ID

示例

请求示例

```
/api/v2/projects/project1
```

正常返回示例

`XML` 格式

```
<requestId>3F4ADBCF-B116-4680-80B3-E9A139841A0D</requestId>
```

`JSON` 格式

```
{"requestId":"3F4ADBCF-B116-4680-80B3-E9A139841A0D"}
```

11.3. GetProject

您可以通过GetProject OpenAPI查看指定项目的详细信息。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
GET /api/v2/projects/[projectName] HTTP/1.1
```

请求参数

名称	类型	位置	是否必选	示例值	描述
projectName	String	Path	是	project1	项目名称。
RegionId	String	Header	否	cn-hangzhou	区域ID

返回数据

名称	类型	示例值	描述
Project	Struct		项目详情。
ClusterId	String	rcmp9x37ztfb63g1x7lt****	集群ID。 **说明** 您可以使用listcluster获取集群ID信息。
CreateTime	Long	1597655424000	项目创建时间。
Creator	String	23996037709276****	项目创建者的UID。
DeployType	String	cell	集群类型： • 独享集群：cell。 • 共享集群：public。
Description	String	test	项目描述。
GlobalJobConfig	String	sql.type=print datastream.main.class=test	项目级别系统参数。
Id	String	7ht9ed87rtd48bof9mu8****	请求ID。
ManagerIds	String	107992689699****	管理员ID列表。多个ID使用英文逗号（,）分隔。

名称	类型	示例值	描述
Modifier	String	23996037709276****	项目修改者UID。
ModifyTime	Long	1597655424000	项目修改时间。
Name	String	project1	项目名称。
Region	String	cn-hangzhou	区域ID。
State	String	ENABLE	项目状态。
RequestId	String	4A6BFFEE-6C96-4727-A620-A109F50284DA	请求ID。

示例

请求示例

```
/api/v2/projects/project1
```

正常返回示例

`XML` 格式

```
<Project>
    <DeployType>CELL</DeployType>
    <ModifyTime>1597655424000</ModifyTime>
    <ManagerIds>107992689699****</ManagerIds>
    <Description>test</Description>
    <ClusterId>rcmp9x37ztfb63g1x7lt****</ClusterId>
    <CreateTime>1597655424000</CreateTime>
    <Creator>23996037709276****</Creator>
    <Name>project1</Name>
    <State>ENABLE</State>
    <Region>cn-hangzhou</Region>
    <Id>7ht9ed87rtd48bof9mu8****</Id>
    <Modifier>23996037709276****</Modifier>
</Project>
<RequestId>4A6BFFEE-6C96-4727-A620-A109F50284DA</RequestId>
```

`JSON` 格式

```
{
    "Project": {
        "DeployType": "CELL",
        "ModifyTime": 1597655424000,
        "ManagerIds": "107992689699****",
        "Description": "test",
        "ClusterId": "rcmp9x37ztfb63g1x7lt****",
        "CreateTime": 1597655424000,
        "Creator": "239960377092 76****",
        "Name": "project1",
        "State": "ENABLE",
        "Region": "cn-hangzhou",
        "Id": "7ht9ed87rtd48bof9mu8****",
        "Modifier": "239960377092 76****"
    },
    "RequestId": "4A6BFFEE-6C96-4727-A620-A109F50284DA"
}
```


11.4. ListProject

您可以通过List Project OpenAPI查询多个项目的信息。查询的项目信息可根据参数选择，进行自由组合。

调试

您可以在OpenAPI Explorer中直接运行该接口，免去您计算签名的困扰。运行成功后，OpenAPI Explorer可以自动生成SDK代码示例。

请求头

您可以使用公共请求参数和该接口必填参数。公共请求参数请参见OpenAPI>调用方式>公共请求参数文档。

请求语法

```
GET /api/v2/projects HTTPS
```

请求参数

名称	类型	位置	是否必选	示例值	描述
RegionId	String	Header	否	cn-hangzhou	区域ID
name	String	Query	否	job1	项目名称
deployType	String	Query	否	cell	集群类型： • CELL：独享集群 • PUBLIC：共享集群

名称	类型	位置	是否必选	示例值	描述
clusterId	String	Query	否	cmy99ugusuco66x9qc6k****	集群ID **说明** 您可以使用listcluster获取集群ID信息。
region	String	Query	否	cn-hangzhou	区域ID
pageSize	Integer	Query	否	1	每页显示的项目数
pageIndex	Integer	Query	否	2	查询的页码

返回数据

名称	类型	示例值	描述
PageIndex	Integer	1	查询的页码
PageSize	Integer	10	每页显示的项目数
Projects	Array of Project		项目详情
Project			
ClusterId	String	rcmp9x37ztfb63g1x7lt****	集群ID **说明** 您可以使用listcluster获取集群ID信息。
CreateTime	Long	1597655424000	项目创建时间
Creator	String	23996037709276****	项目创建者的UID
DeployType	String	CELL	集群类型： • CELL：独享集群 • PUBLIC：共享集群

名称	类型	示例值	描述
Description	String	test	项目描述
Id	String	e8edisrtou71a2neroi5f3kc	请求ID
ManagerIds	String	1079926896999421	管理员ID列表。多个ID请使用逗号（,）分隔。
Modifier	String	27395716024545****	项目修改者UID
ModifyTime	Long	1605610424000	项目修改时间
Name	String	project1	项目名称
Region	String	cn-hangzhou	区域ID
State	String	ENABLE	项目状态
RequestId	String	5A0EA261-AA2F-47FF-B7D9-A785ED1D0ADE	请求ID
TotalCount	Long	3	总项目数
TotalPage	Integer	1	总页码数

示例

请求示例

```
/api/v2/projects
```

正常返回示例

`XML` 格式

```
<TotalCount>3</TotalCount>
<TotalPage>1</TotalPage>
<PageSize>10</PageSize>
<RequestId>5A0EA261-AA2F-47FF-B7D9-A785ED1D0ADE</RequestId>
<Projects>
    <Project>
        <DeployType>CELL</DeployType>
        <ModifyTime>1597655424000</ModifyTime>
        <ManagerIds>1079926896999421</ManagerIds>
        <Description>test</Description>
        <ClusterId>rcmp9x37ztfb63g1x7lt****</ClusterId>
        <State>ENABLE</State>
        <CreateTime>1597655424000</CreateTime>
        <Region>cn-hangzhou</Region>
        <Creator>23996037709276****</Creator>
        <Id>7ht9ed87rtd48bof9mu8895z</Id>
        <Modifier>23996037709276****</Modifier>
        <Name>project1</Name>
    </Project>
    <Project>
        <DeployType>CELL</DeployType>
        <ModifyTime>1605610424000</ModifyTime>
        <ManagerIds>1079926896999421</ManagerIds>
        <Description>stream4demo</Description>
        <ClusterId>rulpqd442sgxbtw9736r****</ClusterId>
        <State>ENABLE</State>
        <CreateTime>1605610424000</CreateTime>
        <Region>cn-beijing</Region>
        <Creator>27395716024545****</Creator>
        <Id>e8edisrtou71a2neroi5f3kc</Id>
        <Modifier>27395716024545****</Modifier>
        <Name>stream4demo</Name>
    </Project>
</Projects>
<PageIndex>1</PageIndex>
```

`JSON` 格式

```
{
 "TotalCount": 3,
 "TotalPage": 1,
 "PageSize": 10,
 "RequestId": "5A0EA261-AA2F-47FF-B7D9-A785ED1D0ADE",
 "Projects": {
  "Project": [
   {
    "DeployType": "CELL",
    "ModifyTime": 1597655424000,
    "ManagerIds": "1079926896999421",
    "Description": "test",
    "ClusterId": "rcmp9x37ztfb63g1x7lt****",
    "State": "ENABLE",
    "CreateTime": 1597655424000,
    "Region": "cn-hangzhou",
    "Creator": "23996037709276****",
    "Id": "7ht9ed87rtd48bof9mu8895z",
    "Modifier": "23996037709276****",
    "Name": "project1"
   },
   {
    "DeployType": "CELL",
    "ModifyTime": 1605610424000,
    "ManagerIds": "1079926896999421",
    "Description": "stream4demo",
    "ClusterId": "rulpqd442sgxbtw9736r****",
    "State": "ENABLE",
    "CreateTime": 1605610424000,
    "Region": "cn-beijing",
    "Creator": "27395716024545****",
    "Id": "e8edisrtou71a2neroi5f3kc",
    "Modifier": "27395716024545****",
    "Name": "stream4demo"
   }
  ]
 },
 "PageIndex": 1
}
```

12.OpenAPI DEMO

本文为您介绍公共云环境下，使用实时计算OpenAPI方式操作实时计算作业所需的POM依赖和OpenAPI DEMO。

POM依赖

```
<dependency>
    <groupId>com.aliyun</groupId>
    <artifactId>aliyun-java-sdk-foas</artifactId>
    <version>2.8.0</version>
</dependency>
```

说明 如果需要依赖aliyun-java-sdk-core，请使用4.3.0及以上版本。4.3.0版本依赖示例如下。

```
<dependency>
    <groupId>com.aliyun</groupId>
    <artifactId>aliyun-java-sdk-core</artifactId>
    <version>4.3.0</version>
</dependency>
```

OpenAPI DEMO

```
package sample;
import com.aliyuncs.AcsRequest;
import com.aliyuncs.AcsResponse;
import com.aliyuncs.DefaultAcsClient;
import com.aliyuncs.IAcsClient;
import com.aliyuncs.exceptions.ServerException;
import com.aliyuncs.foas.model.v20181111.*;
import com.aliyuncs.http.FormatType;
import com.aliyuncs.profile.DefaultProfile;
import com.aliyuncs.profile.IClientProfile;
import com.google.gson.Gson;
import java.util.HashMap;
import java.util.Map;
/**
 * 适用于公共云环境。
 *
 * 作业的一般流程是：
 * 如果有依赖package，先创建package。
 * 创建Job->获取planjson->更新planjson到job->上线job->启动job->停止instance。
 * Instance的状态一般如下（流作业为例，括号里的为动作）：
 * Job上线->UNKNOWN[启动job]->WAITING[等待]->RUNNING[暂停]->PAUSED[恢复]->RUNNING[停止]->TERMI
NATED
 *
 * Note:以下示例仅供演示foas接口的调用方式，不保证逻辑上的严谨性，请按照自己的场景重新组织调用逻辑。
 */
public class PublicSample {
    //填写购买的foas所在的regionId，根据您的实际情况修改。
```

```
    private static final String regionId = "cn-shanghai";
    //云账号AK，支持子账号，需要自行保证该账号具有访问实时计算服务的权限（子账号需要得到主账号访问实时计算服务的权限授权）。
    private static final String accessId = "<yourAccessId>";
    private static final String accessKey = "<yourAccessSecret>";
    private static final String projectName = "<yourProjectName>";
    private static final String jobName = "<yourJobName>";
    // 文件夹与Linux文件系统格式相同，以/开始，表示根目录，只支持绝对路径，不支持相对路径。
    private static final String folderName = "/新手任务";
    private static final String clusterId = "<yourClusterId>";//可以通过listProjectBindQueue接口来获得。
    private static final String queueName = "root.<queueName>";//可以通过listProjectBindQueue接口来获得。
    private static final String engineVersion = "current";//Blink或者Flink的版本，current表示使用推荐版本，beta表示使用测试版本。
    private static final String packageName = "<packageName>";
    private static final String sql =
            "--SQL\n" +
                    "--************************************************************************--\n" +
                    "--CreateTime: 2018-12-29 14:35:04\n" +
                    "--Comment: 请输入业务注释信息\n" +
                    "--************************************************************************--\n" +
                    "\n" +
                    "CREATE TABLE datahub_source (`name` INT, num VARCHAR) WITH (\n" +
                    "    type = 'random'\n" +
                    ");\n" +
                    "CREATE TABLE datahub_sink (`name` INT, cnt BIGINT) WITH (\n" +
                    "    type = 'PRINT'\n" +
                    ");\n" +
                    "INSERT INTO\n" +
                    "    datahub_sink\n" +
                    "SELECT\n" +
                    "    `name`,\n" +
                    "    COUNT (1)\n" +
                    "FROM\n" +
                    "    datahub_source\n" +
                    "GROUP BY\n" +
                    "    `name`";
    private static final String properties =
            "#checkpoint模式：EXACTLY_ONCE或者AT_LEAST_ONCE\n" +
                    "blink.checkpoint.mode=EXACTLY_ONCE\n" +
                    "#checkpoint间隔时间，单位毫秒\n" +
                    "blink.checkpoint.interval.ms=180000\n" +
                    "#rocksdb的数据生命周期，单位毫秒\n" +
                    "#state.backend.rocksdb.ttl.ms=129600000\n";
    private static <T extends AcsResponse> T getResponse(IAcsClient client, AcsRequest<T> request) {
        AcsResponse response = null;
        try {
            //必须设置为json。
            request.setHttpContentType(FormatType.JSON);
            request.setAcceptFormat(FormatType.JSON);
            response = client.getAcsResponse(request);
```

```
            response = client.getAcsResponse(request);
        } catch (Exception e) {
            //处理异常的逻辑，请注意异常中的requestId字段，排查问题时，需要提供该值给服务端。
            if (e instanceof ServerException) {
                ServerException serverException = (ServerException) e;
                System.out.println(serverException.getRequestId());//本次请求的requestId。
                System.out.println(serverException.getErrCode());//本次请求失败的错误码，用于排查错误。
            }
            System.out.println(e.getMessage());
        }
        return (T) response;
    }
    private static String getClusterMetricJson(String clusterId) {
        return String.format(
                        "{\n" +
                        "\t\"queries\": [ {\n" +
                        "\t\t\"metric\": \"%s.system.cpu.user\",\n" +
                        "\t\t\"aggregator\": \"avg\",\n" +
                        "\t\t\"granularity\": \"20s\",\n" +
                        "\t\t\"downsample\": \"avg\"\n" +
                        "\t}, {\n" +
                        "\t\t\"metric\": \"%s.system.cpu.iowait\",\n" +
                        "\t\t\"aggregator\": \"avg\",\n" +
                        "\t\t\"granularity\": \"20s\",\n" +
                        "\t\t\"downsample\": \"avg\"\n" +
                        "\t}],\n" +
                        "\t\"start\": %s,\n" +
                        "\t\"end\": %s,\n" +
                        "\t\"limit\": \"avg:sample:50\"\n" +
                        "}",
                clusterId, clusterId, System.currentTimeMillis() - 3600 * 1000, System.currentTimeMillis()
        );
    }
    public static void main(String[] args) throws Exception {
        IClientProfile profile = DefaultProfile.getProfile(regionId, accessId, accessKey);
        DefaultAcsClient client = new DefaultAcsClient(profile);
        client.setAutoRetry(false);//不进行自动重试。
        //创建一个独享集群，该接口是异步接口，调用返回后需要调用listCluster接口来获取集群的状态来判断是否创建完成。
        CreateClusterRequest createClusterRequest = new CreateClusterRequest();
        createClusterRequest.setDescription("test cluster");
        createClusterRequest.setDisplayName("aaaa");
        createClusterRequest.setOrderId("yyyyy");//填写购买独享集群下单后生成的订单ID。
        createClusterRequest.setUserOssBucket("myBucket");
        createClusterRequest.setUserVpcId("xxxxx");
        createClusterRequest.setUserVSwitch("yyyy");
        createClusterRequest.setZoneId("cn-shanghai-f");
        CreateClusterResponse createClusterResponse = PublicSample.getResponse(client, createClusterRequest);
        System.out.println(new Gson().toJson(createClusterResponse));//该response中含有clusterId信息，这个是集群的索引，后面的大部分接口都需要使用到。
        //根据条件搜索集群，不设置的条件留null。
        ListClusterRequest listClusterRequest = new ListClusterRequest();
```

```
        listClusterRequest.setClusterId(clusterId);
        listClusterRequest.setRegion(regionId);
        ListClusterResponse listClusterResponse = PublicSample.getResponse(client, listClus
terRequest);
        System.out.println(new Gson().toJson(listClusterResponse));
        //在该集群上创建queue，只能创建一级queue。
        CreateQueueRequest createQueueRequest = new CreateQueueRequest();
        createQueueRequest.setClusterId(clusterId);
        createQueueRequest.setQueueName(queueName);
        createQueueRequest.setMaxVcore(300);
        createQueueRequest.setMaxMemMB(4 * 1024);
        CreateQueueResponse createQueueResponse = PublicSample.getResponse(client, createQu
eueRequest);
        System.out.println(new Gson().toJson(createQueueResponse));
        //获取集群的引擎版本信息。
        GetClusterEngineVersionsRequest getClusterEngineVersionsRequest = new GetClusterEng
ineVersionsRequest();
        getClusterEngineVersionsRequest.setClusterId(clusterId);
        GetClusterEngineVersionsResponse getClusterEngineVersionsResponse = PublicSample.ge
tResponse(client, getClusterEngineVersionsRequest);
        System.out.println(new Gson().toJson(getClusterEngineVersionsResponse));
        //获取该集群的资源信息。
        GetClusterResourceRequest getClusterResourceRequest = new GetClusterResourceRequest
();
        getClusterResourceRequest.setClusterId(clusterId);
        GetClusterResourceResponse getClusterResourceResponse = PublicSample.getResponse(cl
ient, getClusterResourceRequest);
        System.out.println(new Gson().toJson(getClusterResourceResponse));
        //获取集群的queue信息，包括所有的queue和各个queue的资源信息。
        GetClusterQueueInfoRequest getClusterQueueInfoRequest = new GetClusterQueueInfoRequ
est();
        getClusterQueueInfoRequest.setClusterId(clusterId);
        GetClusterQueueInfoResponse getClusterQueueInfoResponse = PublicSample.getResponse(
client, getClusterQueueInfoRequest);
        System.out.println(new Gson().toJson(getClusterQueueInfoResponse));
        //获取集群的detail信息。
        GetClusterDetailsRequest getClusterDetailsRequest = new GetClusterDetailsRequest();
        getClusterDetailsRequest.setClusterId(clusterId);
        GetClusterDetailsResponse getClusterDetailsResponse = PublicSample.getResponse(clie
nt, getClusterDetailsRequest);
        System.out.println(new Gson().toJson(getClusterDetailsResponse));
        //获取集群的metrics曲线，最多只能获取最近两小时的数据。
        GetClusterMetricsRequest getClusterMetricsRequest = new GetClusterMetricsRequest();
        getClusterMetricsRequest.setClusterId(clusterId);
        getClusterMetricsRequest.setMetricJson(getClusterMetricJson(clusterId));//指标的名称
请查看文档。
        GetClusterMetricsResponse getClusterMetricsResponse = PublicSample.getResponse(clie
nt, getClusterMetricsRequest);
        System.out.println(new Gson().toJson(getClusterMetricsResponse));
        //注意：以下操作独享集群的接口只支持按量付费的独享集群。
        //扩容集群，该接口是异步接口，需要通过集群状态来判断是否操作完成。
        ExpandClusterRequest expandClusterRequest = new ExpandClusterRequest();
        expandClusterRequest.setClusterId(clusterId);
        expandClusterRequest.setModel("Ecs 4c8g");//该参数可选，当前不支持混布，只能填与原有的机
```

```
型一致的机型，或者不填写。
        expandClusterRequest.setCount(20);//目标集群台数。
        ExpandClusterResponse expandClusterResponse = PublicSample.getResponse(client, expa
ndClusterRequest);
        System.out.println(new Gson().toJson(expandClusterResponse));
        //修改集群master的机型配置，只能更改master机型，不能修改master数量。
        //注意：该接口是异步接口，需要通过集群状态来判断是否操作完成。
        ModifyMasterSpecRequest modifyMasterSpecRequest = new ModifyMasterSpecRequest();
        modifyMasterSpecRequest.setClusterId(clusterId);
        modifyMasterSpecRequest.setMasterTargetModel("Ecs_8c16g");
        ModifyMasterSpecResponse modifyMasterSpecResponse = PublicSample.getResponse(client
, modifyMasterSpecRequest);
        System.out.println(new Gson().toJson(modifyMasterSpecResponse));
        //缩容集群，缩容有两种参数提供形式（当前仅支持第一种形式）：
        //1.直接填需要去除的实例instanceId（可以通过getClusterDetails接口获得）。
        //2.提供每种机型的目标台数。
        //注意：该接口是异步接口，需要通过集群状态来判断是否操作完成。
        ShrinkClusterRequest shrinkClusterRequest = new ShrinkClusterRequest();
        shrinkClusterRequest.setClusterId(clusterId);
        shrinkClusterRequest.setInstanceIds("111,222,333");
        //shrinkClusterRequest.setModelTargetCount("{"Ecs_4c8g":2,"Ecs_8c16g":7}");
        ShrinkClusterResponse shrinkClusterResponse = PublicSample.getResponse(client, shri
nkClusterRequest);
        System.out.println(new Gson().toJson(shrinkClusterResponse));
        //销毁集群，只能销毁按量付费的集群。注意：该接口操作风险高，请务必谨慎操作。
        DestroyClusterRequest destroyClusterRequest = new DestroyClusterRequest();
        destroyClusterRequest.setClusterId(clusterId);
        DestroyClusterResponse destroyClusterResponse = PublicSample.getResponse(client, de
stroyClusterRequest);
        System.out.println(new Gson().toJson(destroyClusterResponse));
        //创建project，该接口分两种场景：
        //1.购买的共享集群，需要提供订单ID。
        //2.独享集群上创建project，需要提供独享集群的clusterId。
        CreateProjectRequest createProjectRequest = new CreateProjectRequest();
        //createProjectRequest.setOrderId();//在共享集群上创建project时必填，填写购买的订单id。
        createProjectRequest.setDescription("test project");
        createProjectRequest.setDeployType("cell");//在共享集群上创建project时填public。
        createProjectRequest.setClusterId(clusterId);//在共享集群上创建project时可不填。
        createProjectRequest.setName(projectName);
        CreateProjectResponse createProjectResponse = PublicSample.getResponse(client, crea
teProjectRequest);
        System.out.println(new Gson().toJson(createProjectResponse));
        // 绑定queue到project，该操作类似于将资源（某个queue）指定给某个project使用，只有将资源绑定
到了某个project上，
        // 该project里的作业才能使用该资源。当前只支持一对一的关系，即一个project只能绑定一个queue，
一个queue只能被绑定至一个project上。
        BindQueueRequest bindQueueRequest = new BindQueueRequest();
        bindQueueRequest.setProjectName(projectName);
        bindQueueRequest.setClusterId(clusterId);
        bindQueueRequest.setQueueName(queueName);
        BindQueueResponse bindQueueResponse = PublicSample.getResponse(client, bindQueueReq
uest);
        System.out.println(new Gson().toJson(bindQueueResponse));
        //获取指定project已绑定的资源。
```

```
        ListProjectBindQueueRequest listProjectBindQueueRequest = new ListProjectBindQueueR
equest();
        listProjectBindQueueRequest.setProjectName(projectName);
        ListProjectBindQueueResponse listProjectBindQueueResponse = PublicSample.getRespons
e(client, listProjectBindQueueRequest);
        System.out.println(new Gson().toJson(listProjectBindQueueResponse));
        //获取project详情。
        GetProjectRequest getProjectRequest = new GetProjectRequest();
        getProjectRequest.setProjectName(projectName);
        GetProjectResponse getProjectResponse = PublicSample.getResponse(client, getProject
Request);
        System.out.println(new Gson().toJson(getProjectResponse));
        //根据条件搜索project。
        ListProjectRequest listProjectRequest = new ListProjectRequest();
        listProjectRequest.setName(projectName);
        listProjectRequest.setDeployType("cell");
        listProjectRequest.setRegion("cn-shanghai");
        ListProjectResponse listProjectResponse = PublicSample.getResponse(client, listProj
ectRequest);
        System.out.println(new Gson().toJson(listProjectResponse));
        //从project解绑资源。注意：解绑前，请先将使用了该资源的作业全部下线。
        UnbindQueueRequest unbindQueueRequest = new UnbindQueueRequest();
        unbindQueueRequest.setProjectName(projectName);
        unbindQueueRequest.setClusterId(clusterId);
        unbindQueueRequest.setQueueName(queueName);
        UnbindQueueResponse unbindQueueResponse = PublicSample.getResponse(client, unbindQu
eueRequest);
        System.out.println(new Gson().toJson(unbindQueueResponse));
        //删除project。注意:删除project是高危操作，请务必谨慎操作。删除的条件是该project下的所有作业
均已下线。
        DeleteProjectRequest deleteProjectRequest = new DeleteProjectRequest();
        deleteProjectRequest.setProjectName(projectName);
        DeleteProjectResponse deleteProjectResponse = PublicSample.getResponse(client, dele
teProjectRequest);
        System.out.println(new Gson().toJson(deleteProjectResponse));
        //获取文件夹列表。
        ListChildFolderRequest listChildFolderRequest = new ListChildFolderRequest();
        listChildFolderRequest.setPath("/");
        listChildFolderRequest.setProjectName(projectName);
        ListChildFolderResponse listChildFolderResponse = PublicSample.getResponse(client,
listChildFolderRequest);
        System.out.println(new Gson().toJson(listChildFolderResponse));
        //获取文件夹。
        GetFolderRequest getFolderRequest = new GetFolderRequest();
        getFolderRequest.setPath(folderName);
        getFolderRequest.setProjectName(projectName);
        GetFolderResponse getFolderResponse = PublicSample.getResponse(client, getFolderReq
uest);
        System.out.println(new Gson().toJson(getFolderResponse));
        Long folderId;
        GetFolderResponse.Folder folder = getFolderResponse.getFolder();
        if (folder == null || folder.getFolderId() == null) {
            //创建文件夹。
            CreateFolderRequest createFolderRequest = new CreateFolderRequest();
```

```
            createFolderRequest.setPath(folderName);
            createFolderRequest.setProjectName(projectName);
            CreateFolderResponse createFolderResponse = PublicSample.getResponse(client, cr
eateFolderRequest);
            System.out.println(new Gson().toJson(createFolderResponse));
            folderId = createFolderResponse.getFolderId();
        } else {
            folderId = folder.getFolderId();
        }
        // sdk的Package存在您的OSS上，需要将Package通过OSS console或者OSS sdk上传到OSS上。
        // 此处需要将OSS meta信息提供给foas，
        // 并对foas授权Bucket读取权限(ram role:AliyunStreamDefaultRole)，foas在使用时会从用户指
定的OSS和Bucket上下载Package。
        // 请确保OSS的region与foas所在的region一致。
        // 如果project建在独享集群上，则不需要提供Endpoint，Bucket等信息，只需要提供package的OSSPa
th。
        // 用户的OSS信息在创建独享集群的时候已经提供过了，不需要重复提供。
        CreatePackageRequest createPackageRequest = new CreatePackageRequest();
        createPackageRequest.setProjectName(projectName);
        createPackageRequest.setPackageName(packageName);
        createPackageRequest.setType("jar");
        createPackageRequest.setDescription("test package");
        createPackageRequest.setOssEndpoint("oss-cn-hangzhou.aliyuncs.com");//提供的endpoint
需要保证网络可通达。
        createPackageRequest.setOssBucket("your bucket");
        createPackageRequest.setOssOwner("111111111");//OSS的主账号uid。
        createPackageRequest.setOssPath("aaa/bbb/ccc.jar");
        createPackageRequest.setMd5("123456");//可不填。不填，则跳过md5校验；填写，则在下载时进行
完整性校验。
        CreatePackageResponse createPackageResponse = PublicSample.getResponse(client, crea
tePackageRequest);
        System.out.println(new Gson().toJson(createPackageResponse));
        //更新package。
        UpdatePackageRequest updatePackageRequest = new UpdatePackageRequest();
        updatePackageRequest.setProjectName(projectName);
        updatePackageRequest.setPackageName(packageName);
        updatePackageRequest.setOssPath("aaa/bbb/ddd.jar");
        UpdatePackageResponse updatePackageResponse = PublicSample.getResponse(client, upda
tePackageRequest);
        System.out.println(new Gson().toJson(updatePackageResponse));
        //获取package详情。
        GetPackageRequest getPackageRequest = new GetPackageRequest();
        getPackageRequest.setProjectName(projectName);
        getPackageRequest.setPackageName(packageName);
        GetPackageResponse getPackageResponse = PublicSample.getResponse(client, getPackage
Request);
        System.out.println(new Gson().toJson(getPackageResponse));
        //根据条件搜索package。
        ListPackageRequest listPackageRequest = new ListPackageRequest();
        listPackageRequest.setProjectName(projectName);
        listPackageRequest.setPackageName("aa");//模糊匹配。
        listPackageRequest.setType("jar");
        listPackageRequest.setPageIndex(1);//可选，默认1。
        listPackageRequest.setPageSize(30);//可选，默认10。
```

```
        ListPackageResponse listPackageResponse = PublicSample.getResponse(client, listPack
ageRequest);
        System.out.println(new Gson().toJson(listPackageResponse));
        //列举project绑定的cluster和queue。
        ListProjectBindQueueRequest listProjectBindQueueRequest2 = new ListProjectBindQueue
Request();
        listProjectBindQueueRequest2.setProjectName(projectName);
        ListProjectBindQueueResponse listProjectBindQueueResponse2 = PublicSample.getRespon
se(client, listProjectBindQueueRequest2);
        System.out.println(new Gson().toJson(listProjectBindQueueResponse2));
        //查询project下绑定的queue的资源。
        ListProjectBindQueueResourceRequest listProjectBindQueueResourceRequest = new ListP
rojectBindQueueResourceRequest();
        listProjectBindQueueResourceRequest.setProjectName(projectName);
        ListProjectBindQueueResourceResponse listProjectBindQueueResourceResponse = PublicS
ample.getResponse(client, listProjectBindQueueResourceRequest);
        System.out.println(new Gson().toJson(listProjectBindQueueResourceResponse));
        //根据条件搜索project下的作业。
        ListJobRequest listJobRequest = new ListJobRequest();
        listJobRequest.setProjectName(projectName);
        listJobRequest.setJobType("flink_stream");
        listJobRequest.setApiType("sql");
        //该参数控制返回的job信息中是否返回较长的字段（例如code）。填false，则返回的只有部分字段；不
填或者填true，则返回所有字段。
        listJobRequest.setIsShowFullField(false);
        ListJobResponse listJobResponse = PublicSample.getResponse(client, listJobRequest);
        System.out.println(new Gson().toJson(listJobResponse));
        //获取作业。
        GetJobRequest getJobRequest = new GetJobRequest();
        getJobRequest.setProjectName(projectName);
        getJobRequest.setJobName(jobName);
        GetJobResponse getJobResponse = PublicSample.getResponse(client, getJobRequest);
        System.out.println(new Gson().toJson(getJobResponse));
        GetJobResponse.Job job = getJobResponse.getJob();
        if (job == null || job.getJobName() == null) {
            //创建作业。
            CreateJobRequest createJobRequest = new CreateJobRequest();
            createJobRequest.setProjectName(projectName);
            createJobRequest.setJobName(jobName);
            createJobRequest.setCode(sql);
            createJobRequest.setProperties(properties);//无，则不用设置。
            //clusterId和queueName可以不配置。如果不配置，则随机从项目绑定的cluster和queue中选择一
个。
            createJobRequest.setClusterId(clusterId);
            createJobRequest.setQueueName(queueName);
            //engineVersion可以不配置。如果不配置，则使用current版本（推荐版本）。
            createJobRequest.setEngineVersion(engineVersion);
            createJobRequest.setDescription("test");
            createJobRequest.setPackages(packageName);//多个package之间用英文逗号分隔。
            createJobRequest.setJobType("flink_stream");//flink_stream/flink_batch大小写不敏
感。
            createJobRequest.setApiType("sql");//datastream/sql大小写不敏感。
            createJobRequest.setFolderId(folderId);
            CreateJobResponse createJobResponse = PublicSample.getResponse(client, createJo
bRequest);
```

```
bRequest);
            System.out.println(new Gson().toJson(createJobResponse));
        } else {
            //修改作业。
            UpdateJobRequest updateJobRequest = new UpdateJobRequest();
            updateJobRequest.setProjectName(projectName);
            updateJobRequest.setJobName(jobName);
            updateJobRequest.setCode(sql);//设置需要修改的字段，不修改的字段不需要设置。
            UpdateJobResponse updateJobResponse = PublicSample.getResponse(client, updateJo
bRequest);
            System.out.println(new Gson().toJson(updateJobResponse));
        }
        //返回引用指定package的job，该接口查询上线后的job引用关系，未上线的job不会返回。
        GetRefPackageJobRequest getRefPackageJobRequest = new GetRefPackageJobRequest();
        getRefPackageJobRequest.setProjectName(projectName);
        getRefPackageJobRequest.setPackageName(packageName);
        GetRefPackageJobResponse getRefPackageJobResponse = PublicSample.getResponse(client
, getRefPackageJobRequest);
        System.out.println(new Gson().toJson(getRefPackageJobResponse));
        //校验job是否存在语法错误。
        ValidateJobRequest validateJobRequest = new ValidateJobRequest();
        validateJobRequest.setProjectName(projectName);
        validateJobRequest.setJobName(jobName);
        ValidateJobResponse validateJobResponse = PublicSample.getResponse(client, validate
JobRequest);
        System.out.println(new Gson().toJson(validateJobResponse));
        //获取job的planjson。该接口是异步接口，提交完成之后，需要调用CheckRawPlanJson接口获取结果:S
ession in run/success/fail。
        GetRawPlanJsonRequest getRawPlanJsonRequest = new GetRawPlanJsonRequest();
        getRawPlanJsonRequest.setProjectName(projectName);
        getRawPlanJsonRequest.setJobName(jobName);
        getRawPlanJsonRequest.setAutoconfEnable(true);//可以不填，默认为false。
        //设置这个job预期使用多少资源来运行，这个参数会影响到生成plan的大小，不提供的话，底层引擎会生
成默认资源配置。
        getRawPlanJsonRequest.setExpectedCore(2f);//这个参数与下面的参数可以同时提供，也可以同时
不提供，不允许一个提供，一个不提供。
        getRawPlanJsonRequest.setExpectedGB(9f);
        GetRawPlanJsonResponse getRawPlanJsonResponse = PublicSample.getResponse(client, ge
tRawPlanJsonRequest);
        CheckRawPlanJsonRequest checkRawPlanJsonRequest = new CheckRawPlanJsonRequest();
        checkRawPlanJsonRequest.setProjectName(projectName);
        checkRawPlanJsonRequest.setJobName(jobName);
        checkRawPlanJsonRequest.setSessionId(getRawPlanJsonResponse.getSessionId());
        String planJson;
        CheckRawPlanJsonResponse checkRawPlanJsonResponse;
        while (true) {
            checkRawPlanJsonResponse = PublicSample.getResponse(client, checkRawPlanJsonReq
uest);
            System.out.println(checkRawPlanJsonResponse.getPlanJsonInfo().getStatus());
            if (checkRawPlanJsonResponse.getPlanJsonInfo().getStatus().equals("success")) {
                planJson = checkRawPlanJsonResponse.getPlanJsonInfo().getPlanJson();
                break;
            } else if (checkRawPlanJsonResponse.getPlanJsonInfo().getStatus().equals("fail"
)) {
                System.out.println(checkRawPlanJsonResponse.getPlanJsonInfo().getErrorMessa
```

```
ge());
                return;
            }
            Thread.sleep(1000);//每秒查询一次。
        }
        //修改作业planjson。
        UpdateJobRequest updateJobRequest = new UpdateJobRequest();
        updateJobRequest.setProjectName(projectName);
        updateJobRequest.setJobName(jobName);
        updateJobRequest.setPlanJson(planJson);
        UpdateJobResponse updateJobResponse = PublicSample.getResponse(client, updateJobReq
uest);
        System.out.println(new Gson().toJson(updateJobResponse));
        //预估当前job所配置的planjson需要的资源数，为上线接口需要提供的maxCU数值提供参考，仅blink-3
.2.1及以上版本支持该功能。
        CalcPlanJsonResourceRequest calcPlanJsonResourceRequest = new CalcPlanJsonResourceR
equest();
        calcPlanJsonResourceRequest.setProjectName(projectName);
        calcPlanJsonResourceRequest.setJobName(jobName);
        CalcPlanJsonResourceResponse calcPlanJsonResourceResponse = PublicSample.getRespons
e(client, calcPlanJsonResourceRequest);
        System.out.println(new Gson().toJson(calcPlanJsonResourceResponse));
        //获取作业上一次运行记录的最后一次自动调优的planjson，注意:是上一次的运行记录，包括停止的和暂
停的。如果不存在，则返回null。
        GetJobLatestAutoScalePlanRequest getJobLatestAutoScalePlanRequest = new GetJobLates
tAutoScalePlanRequest();
        getJobLatestAutoScalePlanRequest.setProjectName(projectName);
        getJobLatestAutoScalePlanRequest.setJobName(jobName);
        GetJobLatestAutoScalePlanResponse getJobLatestAutoScalePlanResponse = PublicSample.
getResponse(client, getJobLatestAutoScalePlanRequest);
        System.out.println(new Gson().toJson(getJobLatestAutoScalePlanResponse));
        //上线作业。
        CommitJobRequest commitJobRequest = new CommitJobRequest();
        commitJobRequest.setProjectName(projectName);
        commitJobRequest.setJobName(jobName);
        commitJobRequest.setIsOnOff(true);//是否开启autoscale自动调优。不填，表示不开启，仅blink
-3.2.1及以上版本支持该功能。
        commitJobRequest.setMaxCU(100);//设置最大资源使用上限。
        commitJobRequest.setConfigure("{\"fetch_delay\":44.5}");//json格式，当前仅支持这一个配
置项，大小写敏感，value是double型。
        CommitJobResponse commitJobResponse = PublicSample.getResponse(client, commitJobReq
uest);
        System.out.println(new Gson().toJson(commitJobResponse));
        //获取实例。
        GetInstanceRequest getInstanceRequest = new GetInstanceRequest();
        getInstanceRequest.setProjectName(projectName);
        getInstanceRequest.setJobName(jobName);
        //-1表示获取流作业的当前运行实例（因为流作业某一时刻最多只会存在一个示例）。注意：批作业需要填
实际的运行实例id。
        getInstanceRequest.setInstanceId(-1L);
        GetInstanceResponse getInstanceResponse = PublicSample.getResponse(client, getInsta
nceRequest);
        System.out.println(new Gson().toJson(getInstanceResponse));
        GetInstanceResponse.Instance instance = getInstanceResponse.getInstance();
```

```
        {
            //启动作业。
            if (instance == null || "UNKNOWN".equals(instance.getActualState()) || "TERMINA
TED".equals(instance.getActualState())) {
                StartJobRequest startJobRequest = new StartJobRequest();
                startJobRequest.setProjectName(projectName);
                startJobRequest.setJobName(jobName);
                Map map = new HashMap<String, String>();
                //startOffset表示启动点位。
                map.put("startOffset", String.valueOf(System.currentTimeMillis()));//精确到m
s时间戳。
                startJobRequest.setParameterJson(new Gson().toJson(map));//json格式参数。
                StartJobResponse startJobResponse = PublicSample.getResponse(client, startJ
obRequest);
                System.out.println(new Gson().toJson(startJobResponse));
                //等待启动成功。
                while (true) {
                    GetInstanceRunSummaryRequest getInstanceRunSummaryRequest = new GetInst
anceRunSummaryRequest();
                    getInstanceRunSummaryRequest.setProjectName(projectName);
                    getInstanceRunSummaryRequest.setJobName(jobName);
                    getInstanceRunSummaryRequest.setInstanceId(-1L);
                    GetInstanceRunSummaryResponse getInstanceRunSummaryResponse = PublicSam
ple.getResponse(client, getInstanceRunSummaryRequest);
                    System.out.println(new Gson().toJson(getInstanceRunSummaryResponse));
                    if ("WAITING".equals(getInstanceRunSummaryResponse.getRunSummary().getA
ctualState())) {
                        System.out.println(String.format("lastErrorTime[%s], lastErrorMessa
ge[%s]",
                                getInstanceRunSummaryResponse.getRunSummary().getLastErrorT
ime(),
                                getInstanceRunSummaryResponse.getRunSummary().getLastErrorM
essage()));
                        Thread.sleep(1000);
                    } else {
                        break;
                    }
                }
            }
        }
        {
            // 作业启动成功后，可以调用如下的接口进行运维。
            //获取instance的运行dag图。
            GetInstanceDetailRequest getInstanceDetailRequest = new GetInstanceDetailReques
t();
            getInstanceDetailRequest.setProjectName(projectName);
            getInstanceDetailRequest.setJobName(jobName);
            getInstanceDetailRequest.setInstanceId(-1L);
            GetInstanceDetailResponse getInstanceDetailResponse = PublicSample.getResponse(
client, getInstanceDetailRequest);
            System.out.println(new Gson().toJson(getInstanceDetailResponse));
            //获取instance的运行时配置。
            GetInstanceConfigRequest getInstanceConfigRequest = new GetInstanceConfigReques
t();
```

```
            getInstanceConfigRequest.setProjectName(projectName);
            getInstanceConfigRequest.setJobName(jobName);
            getInstanceConfigRequest.setInstanceId(-1L);
            GetInstanceConfigResponse getInstanceConfigResponse = PublicSample.getResponse(
client, getInstanceConfigRequest);
            System.out.println(new Gson().toJson(getInstanceConfigResponse));
            //获取instance的运行时checkpoint信息。
            GetInstanceCheckpointRequest getInstanceCheckpointRequest = new GetInstanceChec
kpointRequest();
            getInstanceCheckpointRequest.setProjectName(projectName);
            getInstanceCheckpointRequest.setJobName(jobName);
            getInstanceCheckpointRequest.setInstanceId(-1L);
            GetInstanceCheckpointResponse getInstanceCheckpointResponse = PublicSample.getR
esponse(client, getInstanceCheckpointRequest);
            System.out.println(new Gson().toJson(getInstanceCheckpointResponse));
            //获取instance的运行时异常信息(failover信息）。
            GetInstanceExceptionsRequest getInstanceExceptionsRequest = new GetInstanceExce
ptionsRequest();
            getInstanceExceptionsRequest.setProjectName(projectName);
            getInstanceExceptionsRequest.setJobName(jobName);
            getInstanceExceptionsRequest.setInstanceId(-1L);
            GetInstanceExceptionsResponse getInstanceExceptionsResponse = PublicSample.getR
esponse(client, getInstanceExceptionsRequest);
            System.out.println(new Gson().toJson(getInstanceExceptionsResponse));
            //如果上线时作业开启了自动调优，调优效果不理想，可中途关闭自动调优，关闭之后也可以再次打开
。
            UpdateAutoScaleConfigRequest updateAutoScaleConfigRequest = new UpdateAutoScale
ConfigRequest();
            updateAutoScaleConfigRequest.setProjectName(projectName);
            updateAutoScaleConfigRequest.setJobName(jobName);
            updateAutoScaleConfigRequest.setInstanceId(-1L);
            updateAutoScaleConfigRequest.setConfigJson("{\"autoscale.enable\":false}");//js
on格式，当前只支持这个key，大小写敏感，取值为true或者false。
            UpdateAutoScaleConfigResponse updateAutoScaleConfigResponse = PublicSample.getR
esponse(client, updateAutoScaleConfigRequest);
            System.out.println(new Gson().toJson(updateAutoScaleConfigResponse));
            //查询instance的运行时各项指标的数据曲线信息。
            //当前只支持查询最近两小时的曲线，metric的格式为：blink.{projectName}.{jobName}.{met
ricName}。
            String metricJson = "{\n" +
                    "\t\"start\": 1547637620000,\n" + //精确到ms时间戳。
                    "\t\"limit\": \"avg:sample:50\",\n" +
                    "\t\"end\": 1547638420000,\n" +
                    "\t\"queries\": [{\n" +
                    "\t\t\"downsample\": \"20s-avg\",\n" +
                    "\t\t\"metric\": \"blink.bayes_team.huayuan_test_job.delay\",\n" +
                    "\t\t\"granularity\": \"20s\",\n" +
                    "\t\t\"aggregator\": \"max\"\n" +
                    "\t}, {\n" +
                    "\t\t\"downsample\": \"20s-avg\",\n" +
                    "\t\t\"metric\": \"blink.bayes_team.huayuan_test_job.fetched_delay\",\n
" +
                    "\t\t\"granularity\": \"20s\",\n" +
                    "\t\t\"aggregator\": \"max\"\n" +
```

```
                    "\t}]\n" +
                    "}";
            GetInstanceMetricRequest getInstanceMetricRequest = new GetInstanceMetricReques
t();
            getInstanceMetricRequest.setProjectName(projectName);
            getInstanceMetricRequest.setJobName(jobName);
            getInstanceMetricRequest.setInstanceId(-1L);
            getInstanceMetricRequest.setMetricJson(metricJson);
            GetInstanceMetricResponse getInstanceMetricResponse = PublicSample.getResponse(
client, getInstanceMetricRequest);
            System.out.println(new Gson().toJson(getInstanceMetricResponse.getMetrics()));
            //获取instance实际使用的资源信息（cpu/memory）。
            GetInstanceResourceRequest getInstanceResourceRequest = new GetInstanceResource
Request();
            getInstanceResourceRequest.setProjectName(projectName);
            getInstanceResourceRequest.setJobName(jobName);
            getInstanceResourceRequest.setInstanceId(-1L);
            GetInstanceResourceResponse getInstanceResourceResponse = PublicSample.getRespo
nse(client, getInstanceResourceRequest);
            System.out.println(new Gson().toJson(getInstanceResourceResponse.getResource())
);
            //批量获取多个作业的运行时状态。
            BatchGetInstanceRunSummaryRequest batchGetInstanceRunSummaryRequest = new Batch
GetInstanceRunSummaryRequest();
            batchGetInstanceRunSummaryRequest.setProjectName(projectName);
            batchGetInstanceRunSummaryRequest.setJobType("flink_stream");
            batchGetInstanceRunSummaryRequest.setJobNames("job1,job2,job3");//多个job之间用英
文逗号分隔，请确保都是stream job，并且job已存在且已上线，否则会忽略错误的job。
            BatchGetInstanceRunSummaryResponse batchGetInstanceRunSummaryResponse = PublicS
ample.getResponse(client, batchGetInstanceRunSummaryRequest);
            System.out.println(new Gson().toJson(batchGetInstanceRunSummaryResponse.getRunS
ummarys()));
        }
        {
            //暂停作业。
            GetInstanceRunSummaryRequest getInstanceRunSummaryRequest = new GetInstanceRunS
ummaryRequest();
            getInstanceRunSummaryRequest.setProjectName(projectName);
            getInstanceRunSummaryRequest.setJobName(jobName);
            getInstanceRunSummaryRequest.setInstanceId(-1L);
            GetInstanceRunSummaryResponse getInstanceRunSummaryResponse = PublicSample.getR
esponse(client, getInstanceRunSummaryRequest);
            System.out.println(new Gson().toJson(getInstanceRunSummaryResponse));
            if ("RUNNING".equals(getInstanceRunSummaryResponse.getRunSummary().getActualSta
te())) {
                ModifyInstanceStateRequest modifyInstanceStateRequest = new ModifyInstanceS
tateRequest();
                modifyInstanceStateRequest.setProjectName(projectName);
                modifyInstanceStateRequest.setJobName(jobName);
                modifyInstanceStateRequest.setInstanceId(-1L);
                modifyInstanceStateRequest.setExpectState("PAUSED");
                ModifyInstanceStateResponse modifyInstanceStateResponse = PublicSample.getR
esponse(client, modifyInstanceStateRequest);
                System.out.println(new Gson().toJson(modifyInstanceStateResponse));
                //等待暂停成功
```

```
                //等待暂停成功。
                while (true) {
                    getInstanceRunSummaryRequest = new GetInstanceRunSummaryRequest();
                    getInstanceRunSummaryRequest.setProjectName(projectName);
                    getInstanceRunSummaryRequest.setJobName(jobName);
                    getInstanceRunSummaryRequest.setInstanceId(-1L);
                    getInstanceRunSummaryResponse = PublicSample.getResponse(client, getIns
tanceRunSummaryRequest);
                    System.out.println(new Gson().toJson(getInstanceRunSummaryResponse));
                    if ("RUNNING".equals(getInstanceRunSummaryResponse.getRunSummary().getA
ctualState())) {
                        System.out.println(String.format("lastErrorTime[%s], lastErrorMessa
ge[%s]",
                                getInstanceRunSummaryResponse.getRunSummary().getLastErrorT
ime(),
                                getInstanceRunSummaryResponse.getRunSummary().getLastErrorM
essage()));
                        Thread.sleep(1000);
                    } else {
                        break;
                    }
                }
            }
        }
        {
            //恢复作业。
            GetInstanceRunSummaryRequest getInstanceRunSummaryRequest = new GetInstanceRunS
ummaryRequest();
            getInstanceRunSummaryRequest.setProjectName(projectName);
            getInstanceRunSummaryRequest.setJobName(jobName);
            getInstanceRunSummaryRequest.setInstanceId(-1L);
            GetInstanceRunSummaryResponse getInstanceRunSummaryResponse = PublicSample.getR
esponse(client, getInstanceRunSummaryRequest);
            System.out.println(new Gson().toJson(getInstanceRunSummaryResponse));
            if ("PAUSED".equals(getInstanceRunSummaryResponse.getRunSummary().getActualStat
e())) {
                ModifyInstanceStateRequest modifyInstanceStateRequest = new ModifyInstanceS
tateRequest();
                modifyInstanceStateRequest.setProjectName(projectName);
                modifyInstanceStateRequest.setJobName(jobName);
                modifyInstanceStateRequest.setInstanceId(-1L);
                modifyInstanceStateRequest.setExpectState("RUNNING");
                ModifyInstanceStateResponse modifyInstanceStateResponse = PublicSample.getR
esponse(client, modifyInstanceStateRequest);
                System.out.println(new Gson().toJson(modifyInstanceStateResponse));
                //等待恢复成功。
                while (true) {
                    getInstanceRunSummaryRequest = new GetInstanceRunSummaryRequest();
                    getInstanceRunSummaryRequest.setProjectName(projectName);
                    getInstanceRunSummaryRequest.setJobName(jobName);
                    getInstanceRunSummaryRequest.setInstanceId(-1L);
                    getInstanceRunSummaryResponse = PublicSample.getResponse(client, getIns
tanceRunSummaryRequest);
                    System.out.println(new Gson().toJson(getInstanceRunSummaryResponse));
                    if ("PAUSED".equals(getInstanceRunSummaryResponse.getRunSummary().getAc
```

```
tualState())) {
                        System.out.println(String.format("lastErrorTime[%s], lastErrorMessa
ge[%s]",
                                getInstanceRunSummaryResponse.getRunSummary().getLastErrorT
ime(),
                                getInstanceRunSummaryResponse.getRunSummary().getLastErrorM
essage()));
                        Thread.sleep(1000);
                    } else {
                        break;
                    }
                }
            }
        }
        {
            //停止作业。
            GetInstanceRunSummaryRequest getInstanceRunSummaryRequest = new GetInstanceRunS
ummaryRequest();
            getInstanceRunSummaryRequest.setProjectName(projectName);
            getInstanceRunSummaryRequest.setJobName(jobName);
            getInstanceRunSummaryRequest.setInstanceId(-1L);
            GetInstanceRunSummaryResponse getInstanceRunSummaryResponse = PublicSample.getR
esponse(client, getInstanceRunSummaryRequest);
            System.out.println(new Gson().toJson(getInstanceRunSummaryResponse));
            if ("RUNNING".equals(getInstanceRunSummaryResponse.getRunSummary().getActualSta
te())) {
                ModifyInstanceStateRequest modifyInstanceStateRequest = new ModifyInstanceS
tateRequest();
                modifyInstanceStateRequest.setProjectName(projectName);
                modifyInstanceStateRequest.setJobName(jobName);
                modifyInstanceStateRequest.setInstanceId(-1L);
                modifyInstanceStateRequest.setExpectState("TERMINATED");
                ModifyInstanceStateResponse modifyInstanceStateResponse = PublicSample.getR
esponse(client, modifyInstanceStateRequest);
                System.out.println(new Gson().toJson(modifyInstanceStateResponse));
                //等待停止成功。
                while (true) {
                    getInstanceRunSummaryRequest = new GetInstanceRunSummaryRequest();
                    getInstanceRunSummaryRequest.setProjectName(projectName);
                    getInstanceRunSummaryRequest.setJobName(jobName);
                    getInstanceRunSummaryRequest.setInstanceId(-1L);
                    getInstanceRunSummaryResponse = PublicSample.getResponse(client, getIns
tanceRunSummaryRequest);
                    System.out.println(new Gson().toJson(getInstanceRunSummaryResponse));
                    if ("RUNNING".equals(getInstanceRunSummaryResponse.getRunSummary().getA
ctualState())) {
                        System.out.println(String.format("lastErrorTime[%s], lastErrorMessa
ge[%s]",
                                getInstanceRunSummaryResponse.getRunSummary().getLastErrorT
ime(),
                                getInstanceRunSummaryResponse.getRunSummary().getLastErrorM
essage()));
                        Thread.sleep(1000);
                    } else {
```

```
                        break;
                    }
                }
            }
        }
        //下线作业。
        OfflineJobRequest offlineJobRequest = new OfflineJobRequest();
        offlineJobRequest.setProjectName(projectName);
        offlineJobRequest.setJobName(jobName);
        OfflineJobResponse offlineJobResponse = PublicSample.getResponse(client, offlineJob
Request);
        System.out.println(new Gson().toJson(offlineJobResponse));
        //删除作业。
        DeleteJobRequest deleteJobRequest = new DeleteJobRequest();
        deleteJobRequest.setProjectName(projectName);
        deleteJobRequest.setJobName(jobName);
        DeleteJobResponse deleteJobResponse = PublicSample.getResponse(client, deleteJobReq
uest);
        System.out.println(new Gson().toJson(deleteJobResponse));
        //删除package。
        DeletePackageRequest deletePackageRequest = new DeletePackageRequest();
        deletePackageRequest.setProjectName(projectName);
        deletePackageRequest.setPackageName(packageName);
        DeletePackageResponse deletePackageResponse = PublicSample.getResponse(client, dele
tePackageRequest);
        System.out.println(new Gson().toJson(deletePackageResponse));
    }
}
```