ALIBABA CLOUD

聚石塔

聚石塔
聚石塔API参考

文档版本：20231221

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
警告	适用于可能会产生风险的场景。介绍用户在操作前就必须充分了解的信息、操作前必须要注意的事项或已具备的条件。	警告 重启操作将导致业务短暂中断，建议您在业务低峰期执行重启操作，或确保已完成数据备份。如有必要，请联系阿里云技术支持提供协助。
重要	在操作前需要用户了解的提示信息、补充信息、注意事项、限制信息等。	重要 再次登录系统时，您需要修改登录账户的初始密码。
说明	用于额外的补充说明、最佳实践、窍门等，不是用户必须了解的信息。	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.应用管理

1.1. 应用基础信息

1.1.1. CreateApp

调用CreateApp 创建一个云应用。

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	CreateApp	系统规定参数。取值：CreateApp。
BizCode	String	是	JST	业务域限制 JST：聚石塔。 NEW_RETAIL：线下零售。 MINI_APP：轻应用云。 SUPPLY：供应链。 MESSAGE：消息业务。
Language	String	是	Java	只支持的编程语言 (Java，PHP，C#，Python)
OperatingSystem	String	是	Linux	支持的部署操作系统 Linux，Windows
ServiceType	String	是	StoreApplication	应用类型 GeneralApplication：其他云应用 StoreApplication：电商云应用 TaoHuDongApplication：小程序云应用
Title	String	是	mmw_test	应用标题(唯一)
UserRoles.N.RoleName	String	是	Owner	用户角色名称。可取值（大小写敏感）： 1. Owner 2. PE 3. Dev 4. Test

UserRoles.N.UserId	String	是	123456	用户ID。可通过用户相关接口获取
UserRoles.N.UserType	String	是	TAOBAO	用户类型，目前有两种：DING_TALK, TAOBAO
BizTitle	String	否	技术博客	应用业务标题
Description	String	否	描述	应用描述
StateType	Integer	否	1	应用状态类型 （默认为无状态应用，1：无状态应用，2：有状态应用， 3：守护进程集， 5：定时任务）
Namespace	String	否	test-namespace	用户自定义k8s的namespace
GroupName	String	否	test-group	创建应用时，直接指定分组名称，会直接绑定到指定的分组。非必填。
MiddleWareIdList.N	RepeatList	否	1	中间件id列表，非必填

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功
ErrMsg	String	xx	错误信息
RequestId	String	C181375C-xxx-xxxx-8D9B-4F723D3FFF1E	请求id
Result	Struct		返回结果
AppId	Long	0000	创建成功的应用id

示例

请求示例

```
http(s)://[Endpoint]/?Action=CreateApp
&BizCode=JST
&Language=Java
&OperatingSystem=Linux
&ServiceType=StoreApplication
&Title=mmw_test
&UserRoles.1.RoleName=Owner
&UserRoles.1.UserId=123456
&UserRoles.1.UserType=TAOBAO
&<公共请求参数>
```

正常返回示例

`XML` **格式**

```
<CreateAppResponse>
  <Result>
        <AppId>0</AppId>
  </Result>
  <RequestId>BACE3320-xxxx-4A60-9CC7-6DBA993E3E9A</RequestId>
  <Code>0</Code>
</CreateAppResponse>
```

`JSON` **格式**

```
{
    "Result": {
        "AppId": 0
    },
    "RequestId": "BACE3320-xxxx-4A60-9CC7-6DBA993E3E9A",
    "Code": 0
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

1.1.2. UpdateApp

调用UpdateApp修改app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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	UpdateApp	系统规定参数。取值：UpdateApp。

AppId	Long	是	0000	appid 应用唯一标识
UserRoles.N.RoleName	String	是	PE	用户角色名称。目前有4中：Owner，PE，Dev，Test
UserRoles.N.UserId	String	是	12345	用户ID
UserRoles.N.UserType	String	是	TAOBAO	用户类型，可取值：DING_TALK, TAOBAO
BizTitle	String	否	博客应用	应用业务描述
Description	String	否	描述	应用描述
Language	String	否	Java	程序语言设计（Java,PHP,C#,Python）
OperatingSystem	String	否	Linux	应用部署系统（Linux，Windows）
ServiceType	String	否	GeneralApplication	应用类型（GeneralApplication：其他云应用。StoreApplication：电商云应用。TaoHuDongApplication：小程序云应用）

返回数据

名称	类型	示例值	描述
RequestId	String	FD0E1170-xxxx-xxxx-8D2F-0E32F904F169	请求id
Code	Integer	0	返回码，0表示成功
ErrMsg	String	xx	错误信息
Result	Struct		结果
Success	Boolean	true	是否成功

示例

请求示例

```
http(s)://[Endpoint]/?Action=UpdateApp
&AppId=0000
&<公共请求参数>
```

正常返回示例

`XML` **格式**

```
<UpdateAppResponse>
  <Result>
        <Success>true</Success>
  </Result>
  <ErrMsg></ErrMsg>
  <RequestId>FD0E1170-xxxx-xxxx-8D2F-0E32F904F169</RequestId>
  <Code>0</Code>
</UpdateAppResponse>
```

`JSON` **格式**

```
{
  "Result": {
    "Success": true
  },
  "ErrMsg": "",
  "RequestId": "FD0E1170-xxxx-xxxx-8D2F-0E32F904F169",
  "Code": 0
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

1.1.3. ListApp

调用ListApp获取人员下app信息集合。

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	ListApp	系统规定参数。取值：ListApp。
PageNumber	Integer	否	1	当前页
PageSize	Integer	否	20	每页条数（默认20）

返回数据

名称	类型	示例值	描述
RequestId	String	12501766-xxxx-xxxx-9D67-C471809D879C	请求id
Code	Integer	0	返回码，0表示成功
ErrorMsg	String	xx	错误信息
TotalCount	Integer	20	总数
Data	Array		查询结果数据
AppId	Long	0000	应用id
Title	String	mmw	应用名称
Description	String	描述	描述
Language	String	Java	编程语言
OperatingSystem	String	Linux	操作系统
BizName	String	JST	业务域
ServiceType	String	StoreApplication	服务类型
DeployType	String	VMDeployment	部署类型
BizTitle	String	博客应用	别名
AppStateType	String	stateless	stateless: 无状态应用, stateful: 有状态应用

示例

请求示例

```
http(s)://[Endpoint]/?Action=ListApp
&<公共请求参数>
```

正常返回示例

`XML` **格式**

```
<ListAppResponse>
      <Data>
            <Description></Description>
            <DeployType>VMDeployment</DeployType>
            <AppStateType>stateless</AppStateType>
            <BizName>JST</BizName>
            <ServiceType>GeneralApplication</ServiceType>
            <AppId>1234</AppId>
            <Language>Java</Language>
            <BizTitle></BizTitle>
            <OperatingSystem>Linux</OperatingSystem>
            <Title>mmw_asdsd</Title>
      </Data>
      <Data>
            <DeployType>VMDeployment</DeployType>
            <Description>马明伟测试pop</Description>
            <AppStateType>stateless</AppStateType>
            <BizName>JST</BizName>
            <ServiceType>GeneralApplication</ServiceType>
            <AppId>5678</AppId>
            <Language>C#</Language>
            <BizTitle>testbIztitle</BizTitle>
            <OperatingSystem>Windows</OperatingSystem>
            <Title>mmw_test1</Title>
      </Data>
      <TotalCount>5</TotalCount>
      <RequestId>516AB655-xxxx-4807-xxxx-4577027D7D1C</RequestId>
      <Code>0</Code>
</ListAppResponse>
```

`JSON` **格式**

```
{
  "Data": [
    {
      "Description": "",
      "DeployType": "VMDeployment",
      "AppStateType": "stateless",
      "BizName": "JST",
      "ServiceType": "GeneralApplication",
      "AppId": 1234,
      "Language": "Java",
      "BizTitle": "",
      "OperatingSystem": "Linux",
      "Title": "mmw_asdsd"
    },
    {
      "DeployType": "VMDeployment",
      "Description": "马明伟测试pop",
      "AppStateType": "stateless",
      "BizName": "JST",
      "ServiceType": "GeneralApplication",
      "AppId": 5678,
      "Language": "C#",
      "BizTitle": "testbIztitle",
      "OperatingSystem": "Windows",
      "Title": "mmw_test1"
    }
  ],
  "TotalCount": 5,
  "RequestId": "516AB655-xxxx-4807-xxxx-4577027D7D1C",
  "Code": 0
}
```


错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

1.1.4. DeleteAppDetail

调用DeleteAppDetail删除一个应用

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。

请求参数

名称	类型	是否必选	示例值	描述

Action	String	是	DeleteAppDetail	系统规定参数。取值：DeleteAppDetail。
AppId	Long	否	0	应用id
Force	Boolean	否	false	是否强制删除，若为true，则会删除应用下所有资源，请谨慎操作

返回数据

名称	类型	示例值	描述
RequestId	String	xx	请求id
Code	Integer	0	返回码，0表示成功
ErrMsg	String	xx	错误信息
Result	Struct		返回的结果
Success	Boolean	true	是否成功

示例

请求示例

```
http(s)://[Endpoint]/?Action=DeleteAppDetail
&AppId=0
&Force=false
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<DeleteAppDetailResponse>
      <Result>
            <Success>true</Success>
      </Result>
      <RequestId>xxx-6F44-406A-A436-6823978A6AAA</RequestId>
      <Code>0</Code>
</DeleteAppDetailResponse>
```

`JSON` 格式

```
{
  "Result": {
    "Success": true
  },
  "RequestId": "xxx-xxxx-406A-A436-6823978A6AAA",
  "Code": 0
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

1.1.5. DescribeAppDetail

调用DescribeAppDetail查看一个app应用详细信息。

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	DescribeAppDetail	系统规定参数。取值：DescribeAppDetail。
AppId	Long	是	0000	app应用唯一标识

返回数据

名称	类型	示例值	描述
Code	Long	0	返回码，0表示成功
RequestId	String	2E5D68C6-xxxx-448E-8C9E-2945B537D8F5	请求id
ErrMessage	String	xx	错误信息
Result	Struct		返回结果
ServiceType	String	GeneralApplication	应用类型（"GeneralApplication"， "其他云应用"，"StoreApplication","电商云应用"，"TaoHuDongApplication","小程序云应用"）

BizTitle	String	博客应用	应用别名
BizName	String	JST	业务域
AppId	Long	0	应用id
Language	String	Java	程序编程设计语言（"Java","PHP","C#","Python"）
Title	String	mmwtest	应用名称
OperatingSystem	String	Windows	操作系统（"Linux","Windows"）
DeployType	String	VMDeployment	"VMDeployment","虚拟机部署", "ContainerDeployment","容器部署"
Description	String	miaoshu	应用描述
AppStateType	String	stateless	stateless: 无状态应用, stateful: 有状态应用
UserRoles	Array		当前应用对应的用户角色列表
UserId	String	12345	用户ID
UserType	String	TAOBAO	用户类型，可分为：DING_TALK, TAOBAO
RealName	String	张三	真实姓名
RoleName	String	Owner	用户角色

示例

请求示例

```
http(s)://[Endpoint]/?Action=DescribeAppDetail
&AppId=0000
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<DescribeAppDetailResponse>
      <RequestId>3FA9970E-AF1B-4BE8-BBE4-C340E47E9CE1</RequestId>
      <Code>0</Code>
      <Result>
            <DeployType>ContainerDeployment</DeployType>
            <AppStateType>stateless</AppStateType>
            <OperatingSystem>Linux</OperatingSystem>
            <Description>测试一下pop上线是否正常</Description>
            <AppId>0</AppId>
            <Language>Java</Language>
            <ServiceType>StoreApplication</ServiceType>
            <BizTitle>pop接口线上测试</BizTitle>
            <Title>pop_online_test</Title>
            <BizName>JST</BizName>
      </Result>
</DescribeAppDetailResponse>
```

`JSON` 格式

```
{
  "RequestId": "3FA9970E-AF1B-4BE8-BBE4-C340E47E9CE1",
  "Code": 0,
  "Result": {
    "DeployType": "ContainerDeployment",
    "AppStateType": "stateless",
    "OperatingSystem": "Linux",
    "Description": "测试一下pop上线是否正常",
    "AppId": 0,
    "Language": "Java",
    "ServiceType": "StoreApplication",
    "BizTitle": "pop接口线上测试",
    "Title": "pop_online_test",
    "BizName": "JST"
  }
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

1.2. 环境管理

1.2.1. CreateEnvironment

调用CreateEnvironment创建环境

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	CreateEnvironment	系统规定参数。取值：CreateEnvironment。
AppId	Long	是	0	应用id
AppSchemaId	Long	是	0	应用部署配置id
EnvName	String	是	xx	环境名称
EnvType	Integer	是	1	环境类型: 0:测试;1:正式
Region	String	是	cn-zhangjiakou	阿里云region
Replicas	Integer	是	3	期望的实例副本数
ClusterId	String	否	c1dddxxxxxxx01413cb6fd3xxxxxxxx96	k8s集群id，指定的集群必须是当前用户下、与应用所在业务域相同、与创建环境所指定的环境类型（EnvType）相同。

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功
ErrMsg	String	xxx	错误信息
RequestId	String	xxx	请求id
Result	Struct		创建结果数据
AppEnvId	Long	0	环境id

示例

请求示例

```
http(s)://[Endpoint]/?Action=CreateEnvironment
&AppId=0
&AppSchemaId=0
&EnvName=xx
&EnvType=1
&Region=cn-zhangjiakou
&Replicas=3
&<公共请求参数>
```

正常返回示例

`XML` **格式**

```
<CreateEnvironmentResponse>
      <Result>
            <AppEnvId>0</AppEnvId>
      </Result>
      <RequestId>xxx-CE41-4572-B74B-E96CBC829965</RequestId>
      <Code>0</Code>
</CreateEnvironmentResponse>
```

`JSON` **格式**

```
{
  "Result": {
    "AppEnvId": 0
  },
  "RequestId": "xxx-CE41-4572-B74B-E96CBC829965",
  "Code": 0
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

1.2.2. UpdateEnvironment

调用UpdateEnvironment更新环境

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	UpdateEnvironment	系统规定参数。取值：UpdateEnvironment。

AppEnvId	Long	是	0	应用环境id
AppId	Long	是	0	应用id
AppSchemaId	Long	否	0	应用部署配置id
Replicas	Integer	否	3	期望的实例副本数

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功
RequestId	String	xxx	请求id
ErrMsg	String	xxx	错误信息
Result	Struct		返回结果
Success	Boolean	true	更新是否成功

示例

请求示例

```
http(s)://[Endpoint]/?Action=UpdateEnvironment
&AppEnvId=0
&AppId=0
&AppSchemaId=0
&EnvName=test-env-name
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<UpdateEnvironmentResponse>
      <RequestId>xxxx-91D5-4504-A732-DFB1DD2A1C69</RequestId>
      <Code>0</Code>
      <Result>
            <Success>true</Success>
      </Result>
</UpdateEnvironmentResponse>
```

`JSON` 格式

```
{
  "RequestId": "xxxx-91D5-4504-A732-DFB1DD2A1C69",
  "Code": 0,
  "Result": {
    "Success": true
  }
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

1.2.3. ListAppEnvironment

调用ListAppEnvironment获取应用环境列表

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	ListAppEnvironment	系统规定参数。取值：ListAppEnvironment。
AppId	Long	是	0	应用id
PageNumber	Integer	是	1	页数
PageSize	Integer	是	10	页大小
EnvType	Integer	否	1	环境类型: 0:测试;1:正式
EnvName	String	否	xx	环境名称

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功
Data	Array		返回的数据

AppId	Long	0	应用id
AppSchemaId	Long	0	应用配置id
EnvId	Long	0	环境id
EnvName	String	xx	环境名称
EnvType	Integer	1	环境类型
EnvTypeName	String	正式环境	环境类型名称
Region	String	cn-zhangjiakou	阿里云region
ErrorMsg	String	xx	错误信息
PageNumber	Integer	1	页数
PageSize	Integer	10	页大小
RequestId	String	xx	请求id
TotalCount	Long	5	总条数

示例

请求示例

```
http(s)://[Endpoint]/?Action=ListAppEnvironment
&AppId=0
&EnvName=xx
&EnvType=1
&PageNumber=1
&PageSize=10
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<ListAppEnvironmentResponse>
      <TotalCount>1</TotalCount>
      <PageSize>10</PageSize>
      <RequestId>xxx-0468-4161-89DF-ECCBEC16BE36</RequestId>
      <PageNumber>1</PageNumber>
      <Data>
            <AppId>0</AppId>
            <AppSchemaId>0</AppSchemaId>
            <EnvTypeName>正式</EnvTypeName>
            <EnvId>0</EnvId>
            <Region>cn-zhangjiakou</Region>
            <EnvName>正式环境</EnvName>
            <EnvType>1</EnvType>
      </Data>
      <Code>0</Code>
</ListAppEnvironmentResponse>
```

`JSON` 格式

```
{
  "TotalCount": 1,
  "PageSize": 10,
  "RequestId": "xxx-0468-4161-89DF-ECCBEC16BE36",
  "PageNumber": 1,
  "Data": [
    {
      "AppId": 0,
      "AppSchemaId": 0,
      "EnvTypeName": "正式",
      "EnvId": 0,
      "Region": "cn-zhangjiakou",
      "EnvName": "正式环境",
      "EnvType": 1
    }
  ],
  "Code": 0
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

1.2.4. DescribeAppEnvironmentDetail

调用DescribeAppEnvironmentDetail查询环境详细信息

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	DescribeAppEnvironmentDetail	系统规定参数。取值：DescribeAppEnvironmentDetail。
AppId	Long	是	0	应用id
EnvId	Long	是	0	环境id

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功
ErrMsg	String	xxx	错误信息
RequestId	String	xxx	请求id
Result	Struct		返回结果
AppId	Long	0	应用id
AppSchemaId	Long	0	应用配置id
EnvId	Long	0	环境id
EnvName	String	xx	环境名称
EnvType	Integer	1	环境类型，0:测试；1:正式
EnvTypeName	String	xx	环境类型名称
Region	String	cn-zhangjiakou	阿里云region
Replicas	Integer	2	期望的副本数

示例

请求示例

```
http(s)://[Endpoint]/?Action=DescribeAppEnvironmentDetail
&AppId=0
&EnvId=0
&<公共请求参数>
```

正常返回示例

`XML` **格式**

```
<DescribeAppEnvironmentDetailResponse>
      <Result>
            <AppSchemaId>0</AppSchemaId>
            <EnvId>0</EnvId>
            <Region>cn-zhangjiakou</Region>
            <Replicas>2</Replicas>
            <EnvName>正式环境</EnvName>
            <EnvTypeName>正式</EnvTypeName>
            <AppId>0</AppId>
            <EnvType>1</EnvType>
      </Result>
      <RequestId>xxx-DE9E-480A-BB45-732FEBF9619C</RequestId>
      <Code>0</Code>
</DescribeAppEnvironmentDetailResponse>
```

`JSON` **格式**

```
{
  "Result": {
    "AppSchemaId": 0,
    "EnvId": 0,
    "Region": "cn-zhangjiakou",
    "Replicas": 2,
    "EnvName": "正式环境",
    "EnvTypeName": "正式",
    "AppId": 0,
    "EnvType": 1
  },
  "RequestId": "xxx-DE9E-480A-BB45-732FEBF9619C",
  "Code": 0
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

1.2.5. DeleteAppEnvironment

调用DeleteAppEnvironment删除一个应用环境

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	DeleteAppEnvironment	系统规定参数。取值：DeleteAppEnvironment。
AppId	Long	是	0	应用id
EnvId	Long	是	0	环境id
Force	Boolean	是	false	是否强制删除，true：会删除环境下所有资源，例如应用实例、流量接入等，请谨慎操作，false：会校验是否可删除。

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功
ErrMsg	String	xxx	错误信息
RequestId	String	xxx	请求id
Result	Struct		返回结果
Success	Boolean	true	删除结果

示例

请求示例

```
http(s)://[Endpoint]/?Action=DeleteAppEnvironment
&AppId=0
&EnvId=0
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<DeleteAppEnvironmentResponse>
      <RequestId>xxx-C216-40C0-8252-893B34B2421E</RequestId>
      <Code>0</Code>
      <Result>
            <Success>true</Success>
      </Result>
</DeleteAppEnvironmentResponse>
```

`JSON` 格式

```
{
  "RequestId": "xxx-C216-40C0-8252-893B34B2421E",
  "Code": 0,
  "Result": {
    "Success": true
  }
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

1.2.6. 下线应用环境

下线应用环境，但会保留环境记录本身

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	OfflineAppEnvironment	系统规定参数。取值：**OfflineAppEnvironment**。
AppId	Long	是	123456	应用id
EnvId	Long	是	123456	环境id
DeletePvc	Boolean	否	false	是否删除PVC，未传递默认不删除

返回数据

名称	类型	示例值	描述

Code	Integer	200	CodeEnum
Success	Boolean	true	是否成功
ErrMsg	String	信息提示	信息提示
RequestId	String	xxx-xx-xx	请求id
Result	Object		结果
Success	Boolean	true	success

示例

请求示例

正常返回示例

`JSON` **格式**

```
HTTP/1.1 200 OK
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

1.2.7. 获取应用环境部署基线信息

获取应用环境的部署基线信息

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	DescribeAppEnvDeployBaseline	系统规定参数。取值：**DescribeAppEnvDeployBaseline**。

返回数据

名称	类型	示例值	描述

Code	Integer	200	CodeEnum
ErrMsg	String	信息提示	信息提示
RequestId	String	xxx-xx-xx	请求id
Success	Boolean	true	是否成功
Result	Object		结果
AppId	Long	123456	应用id
EnvId	Long	123456	环境id
CreateTime	String	2022-05-08	创建时间
SchemaId	Long	123456	部署配置schema_id
PacketId	Long	123456	代码包id
PacketUrl	String	https://xxx.xx.com/aa	指定代码包发布时，为代码包Url；纯镜像发布时，为镜像地址
PacketComment	String	描述信息	代码包描述

示例

请求示例

正常返回示例

`JSON` **格式**

```
HTTP/1.1 200 OK
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

1.3. 部署配置管理

1.3.1. CreateDeployConfig

调用CreateDeployConfig创建一个部署配置

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	CreateDeployConfig	系统规定参数。取值：CreateDeployConfig。
AppId	Long	是	0	应用id
EnvType	String	是	online	环境类型，测试：test，正式：online
Name	String	是	xxx	配置名称
CodePath	String	否	/app/code	代码路径，若为空则部署时不需要代码包
ConfigMap	String	否	ConfigMap base64编码	yaml格式的ConfigMap配置文件，需要Base64编码 <note>废弃，由ConfitMapList代替</note>
Deployment	String	否	Deployment base64编码	yaml格式的Deployment配置文件，需要Base64编码，作为无状态应用部署，和StatefulSet不能同时传。
StatefulSet	String	否	StatefulSet base64编码	yaml格式的StatefulSet配置文件，需要Base64编码，作为有状态应用部署，和Deployment不能同时传。
ConfigMapList.N	RepeatList	否	ConfigMap列表，每个都需要base64编码	yaml格式的ConfigMap配置文件列表，每个ConfigMap都需要Base64编码
CronJob	String	否	CronJob base64编码	yaml格式的CronJob配置文件，需要Base64编码，作为定时任务部署，不能不能同时传Deployment和StatefulSet。
SecretList.N	RepeatList	否	（内测中）Secret列表，每个都需要base64编码	<note>内测中</note>yaml格式的Secret列表，每个Secret都需要Base64编码

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功
ErrMessage	String	xxx	错误信息
RequestId	String	xxx	请求id
Result	Struct		返回结果
AppId	Long	0	应用id
Name	String	xx	配置名称
SchemaId	Long	0	配置id

示例

请求示例

```
http(s)://[Endpoint]/?Action=CreateDeployConfig
&AppId=0
&CodePath=/app/code
&EnvType=online
&Name=xxx
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<CreateDeployConfigResponse>
      <ErrMessage></ErrMessage>
      <RequestId>xxxx-1A41-4ED1-B80F-097281B11407</RequestId>
      <Result>
            <Name>正式环境2</Name>
            <SchemaId>0</SchemaId>
            <AppId>0</AppId>
      </Result>
      <Code>0</Code>
</CreateDeployConfigResponse>
```

`JSON` 格式

```
{
  "ErrMessage": "",
  "RequestId": "xxxx-1A41-4ED1-B80F-097281B11407",
  "Result": {
    "Name": "正式环境2",
    "SchemaId": 0,
    "AppId": 0
  },
  "Code": 0
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

1.3.2. ListDeployConfig

调用ListDeployConfig查询部署配置列表

OpenApi接口仅能获取yaml类型的配置，在聚石塔控制台创建的配置，通过OpenApi无法查到。

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	ListDeployConfig	系统规定参数。取值：**ListDeployConfig**。
AppId	Long	是	0	应用id
EnvType	String	否	online	环境类型: test:测试; online:正式
Id	Long	否	0	schemaid，即配置id
Name	String	否	xx	配置名称

返回数据

名称	类型	示例值	描述
RequestId	String	12345dfas	请求id

Code	Integer	0	返回码，0表示成功
ErrorMsg	String	PermissionDeny	错误信息
PageNumber	Integer	1	当前页码
PageSize	Integer	10	每页大小
TotalCount	Long	5	记录总条数
Data	Array of DeployConfigInstance		返回的结果数据
Name	String	xx	部署配置名称
Id	Long	0	部署配置id
AppId	Long	0	应用id
EnvType	String	online	环境类型
ContainerCodePath	Object		代码部署路径
CodePath	String	/code	代码路径
ContainerYamlConf	Object		yaml配置
StatefulSet	String	xx	StatefulSet
Deployment	String	xx	Deployment
CronJob	String	xx	CronJob
ConfigMap	String	xx	ConfigMap <note>废弃，原来该字段的内容，并入到ConfigMapList中了</note>
SecretList	Array of String	xx	SecretList列表

ConfigMapList	Array of String	xx	ConfigMap列表
ContainerResourceRequest	Object		容器资源请求配置
Cpu	String	1	cpu请求核数
Memory	String	1024	内存请求（MB）
Gpu	String	1024	GPU请求
Storage	String	100	存储请求
ContainerResourceLimit	Object		容器资源限制配置
Cpu	String	1	cpu限制核数
Memory	String	1024	内存限制（MB）
Gpu	String	1024	GPU限制
Storage	String	100	存储限制

示例

请求示例

```
http(s)://[Endpoint]/?Action=ListDeployConfig
&AppId=0
&EnvType=online
&Id=0
&Name=xx
&公共请求参数
```

正常返回示例

`XML` 格式

```
HTTP/1.1 200 OK
Content-Type:application/xml

<ListDeployConfigResponse>
    <RequestId>12345dfas</RequestId>
    <Code>0</Code>
    <ErrorMsg>PermissionDeny</ErrorMsg>
    <PageNumber>1</PageNumber>
    <PageSize>10</PageSize>
    <TotalCount>5</TotalCount>
    <Data>
        <Name>xx</Name>
        <Id>0</Id>
        <AppId>0</AppId>
        <EnvType>online</EnvType>
        <ContainerCodePath>
            <CodePath>/code</CodePath>
        </ContainerCodePath>
        <ContainerYamlConf>
            <StatefulSet>xx</StatefulSet>
            <Deployment>xx</Deployment>
            <CronJob>xx</CronJob>
            <ConfigMap>xx</ConfigMap>
            <SecretList>xx</SecretList>
            <ConfigMapList>xx</ConfigMapList>
        </ContainerYamlConf>
        <ContainerResourceRequest>
            <Cpu>1</Cpu>
            <Memory>1024</Memory>
            <Gpu>1024</Gpu>
            <Storage>100</Storage>
        </ContainerResourceRequest>
        <ContainerResourceLimit>
            <Cpu>1</Cpu>
            <Memory>1024</Memory>
            <Gpu>1024</Gpu>
            <Storage>100</Storage>
        </ContainerResourceLimit>
    </Data>
</ListDeployConfigResponse>
```

`JSON` **格式**

```
HTTP/1.1 200 OK
Content-Type:application/json

{
  "RequestId" : "12345dfas",
  "Code" : 0,
  "ErrorMsg" : "PermissionDeny",
  "PageNumber" : 1,
  "PageSize" : 10,
  "TotalCount" : 5,
  "Data" : [ {
    "Name" : "xx",
    "Id" : 0,
    "AppId" : 0,
    "EnvType" : "online",
    "ContainerCodePath" : {
      "CodePath" : "/code"
    },
    "ContainerYamlConf" : {
      "StatefulSet" : "xx",
      "Deployment" : "xx",
      "CronJob" : "xx",
      "ConfigMap" : "xx",
      "SecretList" : [ "xx" ],
      "ConfigMapList" : [ "xx" ]
    },
    "ContainerResourceRequest" : {
      "Cpu" : "1",
      "Memory" : "1024",
      "Gpu" : "1024",
      "Storage" : "100"
    },
    "ContainerResourceLimit" : {
      "Cpu" : "1",
      "Memory" : "1024",
      "Gpu" : "1024",
      "Storage" : "100"
    }
  } ]
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

1.3.3. DeleteDeployConfig

调用DeleteDeployConfig删除一个部署配置

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	DeleteDeployConfig	系统规定参数。取值：DeleteDeployConfig。
SchemaId	Long	是	0	部署配置id

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功
ErrMsg	String	xxx	错误信息
RequestId	String	xxx	请求id
Result	Struct		返回结果
Success	Boolean	true	是否成功

示例

请求示例

```
http(s)://[Endpoint]/?Action=DeleteDeployConfig
&SchemaId=0
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<DeleteDeployConfigResponse>
      <RequestId>xx-2D4E-4F2B-8C7B-4866A791442A</RequestId>
      <Code>0</Code>
      <Result>
            <Success>true</Success>
      </Result>
</DeleteDeployConfigResponse>
```

`JSON` 格式

```
{
  "RequestId": "xx-2D4E-4F2B-8C7B-4866A791442A",
  "Code": 0,
  "Result": {
    "Success": true
  }
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

1.3.4. UpdateDeployConfig

调用UpdateDeployConfig更新部署配置

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	UpdateDeployConfig	系统规定参数。取值：UpdateDeployConfig。
AppId	Long	是	0	应用id
CodePath	String	是	/xxx/xx	代码部署路径，若为空则部署是不需要代码包，若为null则不更新此字段
Id	Long	是	0	部署配置的id
ConfigMap	String	否	xxx	k8s的ConfigMap的原生yaml配置Base64编码后的字符串。 <note>废弃，由ConfigMapList代替</note>
StatefulSet	String	否	xx	k8s的StatefulSet的原生yaml配置Base64编码后的字符串，仅有状态应用支持StatefulSet
Deployment	String	否	xxx	k8s的Deployment的原生yaml配置Base64编码后的字符串，仅无状态应用支持Deployment

ConfigMapList.N	RepeatList	否	xx	k8s的ConfigMap的原生yaml配置列表，每个都需要Base64编码。 <note>更新此字段，会覆盖原来的内容，例如原来有ConfigMap1和ConfigMap2，更新时只传入ConfigMap3，那么该字段就只有ConfigMap3的内容。如需保留原来，则需传原来的ConfigMap1、ConfigMap2，再加上ConfigMap3。若不更新，则会保留原来的ConfigMap1和ConfigMap2。</note>
SecretList.N	RepeatList	否	（内测中）Secret列表，每个都需要base64编码	<note>内测中</note>yaml格式的Secret列表，每个Secret都需要Base64编码
CronJob	String	否	CronJob base64编码	k8s的CronJob的原生yaml配置Base64编码后的字符串，仅定时任务应用支持CronJob

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功
ErrMsg	String	xxx	错误信息
RequestId	String	xxx	请求id
Result	Struct		返回数据
Success	Boolean	true	更新是否成功

示例

请求示例

```
http(s)://[Endpoint]/?Action=UpdateDeployConfig
&AppId=0
&CodePath=/xxx/xx
&Id=0
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<UpdateDeployConfigResponse>
      <Result>
            <Success>true</Success>
      </Result>
      <RequestId>xxx-6F44-406A-A436-6823978A6AAA</RequestId>
      <Code>0</Code>
</UpdateDeployConfigResponse>
```

`JSON` 格式

```
{
  "Result": {
    "Success": true
  },
  "RequestId": "xxx-6F44-406A-A436-6823978A6AAA",
  "Code": 0
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

1.3.5. UpdateNormalDeployConfig

调用UpdateNormalDeployConfig更新应用环境CPU和内存配置，仅适用于非YAML配置。

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	UpdateNormalDeployConfig	系统规定参数。取值：**UpdateNormalDeployConfig**。
ContainerResourceLimit	Object	否		CPU和内存限制值
Cpu	String	否	2	CPU核心数
Memory	String	否	500	内存大小，单位：MB
ContainerResourceRequest	Object	否		CPU和内存请求值
Cpu	String	否	1.5	CPU核心数

Memory	String	否	200	内存大小，单位：MB
AppId	Long	是	35075	应用ID
Id	Long	是	26162	部署配置的id

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功
RequestId	String	79a8ab31-32ce-4d27-a006-339a5eae3b6e	请求id
ErrMsg	String	请求失败：xxx	错误信息
Result	Object		返回数据
Success	Boolean	true	更新是否成功

示例

请求示例

```
http(s)://[Endpoint]/?Action=UpdateNormalDeployConfig
&ContainerResourceLimit={"Cpu":"2","Memory":"500"}
&ContainerResourceRequest={"Cpu":"1.5","Memory":"200"}
&AppId=35075
&Id=26162
&公共请求参数
```

正常返回示例

`XML` 格式

```
HTTP/1.1 200 OK
Content-Type:application/xml

<UpdateNormalDeployConfigResponse>
    <RequestId>79a8ab31-32ce-4d27-a006-339a5eae3b6e</RequestId>
    <Code>0</Code>
    <ErrMsg/>
    <Result>
        <Success>true</Success>
    </Result>
</UpdateNormalDeployConfigResponse>
```

`JSON` 格式

```
HTTP/1.1 200 OK
Content-Type:application/json

{
  "RequestId" : "79a8ab31-32ce-4d27-a006-339a5eae3b6e",
  "Code" : 0,
  "ErrMsg" : "",
  "Result" : {
    "Success" : true
  }
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

1.4. 应用实例

1.4.1. ListAppInstance

调用ListAppInstance查询环境下的应用实例列表

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	ListAppInstance	系统规定参数。取值：ListAppInstance。
AppId	Long	是	0	应用id
EnvId	Long	是	0	环境id
PageNumber	Integer	是	1	页数
PageSize	Integer	是	10	页大小

返回数据

名称	类型	示例值	描述

Code	Integer	0	返回码，0表示成功
TotalCount	Long	5	总条数
ErrMsg	String	xx	错误信息
PageNumber	Integer	1	页数
PageSize	Integer	10	页大小
RequestId	String	xx	请求id
Data	Array of AppInstanceDetail		返回的数据
AppInstanceId	String	xxx	应用实例id
CreateTime	String	2019-12-31T10:01:05	应用实例创建时间
Health	String	RUNNING	健康状态，取值：RUNNING（正常运行中）或ABNORMAL（异常） <note>已废弃，由Status字段代替</note>
HostIp	String	xx.xx.xx.xx	应用实例所在主机ip
Limits	String	1vcpu 2GB	容器资源限制
PodIp	String	xx.xx.xx.xx	应用实例pod的ip
Requests	String	0.5vcpu 1GB	容器资源请求
RestartCount	Integer	0	pod的重启次数
Spec	String	1vcpu 2GB	容器规格 <note>已废弃，由limits代替</note>
Status	String	Normal	实例的状态，Normal：正常，Starting：启动中，ToBeSchedule：待调度，Pending：挂起，Offlining：下线中，Failed：失败，Unknown：未知，Stopped：已停止

Version	String	1.0.0	应用实例的版本，通过镜像发布时，该字段为镜像的版本；通过上传代码包发布时，该字段为创建发布单时填写的版本。

示例

请求示例

```
http(s)://[Endpoint]/?Action=ListAppInstance
&AppId=0
&EnvId=0
&PageNumber=1
&PageSize=10
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<ListAppInstanceResponse>
      <PageNumber>1</PageNumber>
      <Data>
            <PodIp>172.20.5.64</PodIp>
            <AppInstanceId>jck-deployment-yacs-00-00-00-11646-768c4f98b-8st7g</AppInstan
ceId>
            <HostIp>192.168.17.204</HostIp>
            <RestartCount>0</RestartCount>
            <Spec>1vcpu 1GB</Spec>
            <Requests>0.500vcpu 1000MB</Requests>
            <Limits>1vcpu 1GB</Limits>
            <CreateTime>2019-12-31T15:06:20</CreateTime>
            <Health>RUNNING</Health>
            <Version>1.0.0</Version>
            <Status>Normal</Status>
      </Data>
      <TotalCount>1</TotalCount>
      <RequestId>xx-D974-4DBA-A673-2FF0EB8AAA2E</RequestId>
      <Code>0</Code>
</ListAppInstanceResponse>
```

`JSON` 格式

```
{"TotalCount":"5","PageSize":"10","RequestId":"xx","ErrMsg":"xx","PageNumber":"1","Data":
[{"Status":"Normal","HostIp":"xx.xx.xx.xx","RestartCount":"0","AppInstanceId":"xxx","Limi
":"1vcpu
2GB","Version":"1.0.0","Health":"RUNNING","PodIp":"xx.xx.xx.xx","CreateTime":"2019-12-3
1T10:01:05","Spec":"1vcpu 2GB","Requests":"0.5vcpu 1GB"}],"Code":"0"}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

1.4.2. RebuildAppInstance

调用RebuildAppInstance重建应用实例

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	RebuildAppInstance	系统规定参数。取值：RebuildAppInstance。
AppId	Long	是	0	应用Id
AppInstanceId	String	是	jck-deployment-yacs--xxx-xxx-6xx2ftxxx	环境Id
EnvId	Long	是	0	应用实例Id

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功
ErrMsg	String	xxx	错误信息
RequestId	String	xxx	请求Id
Result	Struct		返回结果
Success	Boolean	true	操作是否成功

示例

请求示例

```
http(s)://[Endpoint]/?Action=RebuildAppInstance
&AppId=0
&AppInstanceId=jck-deployment-yacs--xxx-xxx-6xx2ftxxx
&EnvId=0
&<公共请求参数>
```

正常返回示例

`XML` **格式**

```
<RebuildAppInstanceResponse>
      <RequestId>xxx</RequestId>
      <ErrMsg>xxx</ErrMsg>
      <Code>0</Code>
      <Result>
            <Success>true</Success>
      </Result>
</RebuildAppInstanceResponse>
```

`JSON` **格式**

```
{
  "RequestId": "xxx",
  "ErrMsg": "xxx",
  "Code": "0",
  "Result": {
    "Success": "true"
  }
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

1.4.3. RestartAppInstance

通过RestartAppInstance重启应用的容器实例，目前该接口仅支持轻容器应用。

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	RestartAppInstance	系统规定参数。取值：**RestartAppInstance**。
AppId	Long	是	1234	应用ID
EnvId	Long	是	1234	环境ID
AppInstanceIdList.N	Long	是	1234	容器id

返回数据

名称	类型	示例值	描述
RequestId	String	1234xxxxx	requestId
ErrMsg	String	权限校验失败	errMsg
Code	Integer	500	code
Result	String	{"success": true}	result

示例

请求示例

```
http(s)://[Endpoint]/?Action=RestartAppInstance
&AppId=1234
&EnvId=1234
&AppInstanceIdList=[1234]
&公共请求参数
```

正常返回示例

`XML` 格式

```
HTTP/1.1 200 OK
Content-Type:application/xml

<RestartAppInstanceResponse>
    <RequestId>1234xxxxx</RequestId>
    <ErrMsg>权限校验失败</ErrMsg>
    <Code>500</Code>
    <Result>{"success": true}</Result>
</RestartAppInstanceResponse>
```

`JSON` 格式

```
HTTP/1.1 200 OK
Content-Type:application/json

{
  "RequestId" : "1234xxxxx",
  "ErrMsg" : "权限校验失败",
  "Code" : 500,
  "Result" : "{\"success\": true}"
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

1.5. 存储卷声明管理

1.5.1. CreatePersistentVolumeClaim

调用CreatePersistentVolumeClaim创建一个持久化存储卷声明

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	CreatePersistent VolumeClaim	系统规定参数。取值：CreatePersistentVolumeClaim。
AccessModes	String	是	ReadWriteMany	访问模式，可取值：ReadWriteMany、ReadWriteOnce
AppId	Long	是	0	应用id
Capacity	String	是	500Gi	容量大小，支持纯数字、定点整数+以下后缀：E, P, T, G, M, K 、2的幂形式：Ei, Pi, Ti, Gi, Mi, Ki
EnvId	Long	是	0	环境id
Name	String	是	xx	持久化存储卷声明的名称
StorageClass	String	是	xx	存储类名称，名称必须以小写字母开头，只能包含小写字母、数字、"."和"-"

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功
RequestId	String	xx	请求id

ErrMsg	String	xx	错误信息
Result	Struct		返回结果
PersistentVolumeClaimId	Long	0	存储卷声明id

示例

请求示例

```
http(s)://[Endpoint]/?Action=CreatePersistentVolumeClaim
&AccessModes=ReadWriteMany
&AppId=0
&Capacity=500Gi
&EnvId=0
&Name=xx
&StorageClass=xx
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<CreatePersistentVolumeClaimResponse>
      <RequestId>xx-5DB4-427B-88DC-79378EE1990A</RequestId>
      <Code>0</Code>
      <Result>
            <PersistentVolumeClaimId>0</PersistentVolumeClaimId>
      </Result>
</CreatePersistentVolumeClaimResponse>
```

`JSON` 格式

```
{
  "RequestId": "xx-5DB4-427B-88DC-79378EE1990A",
  "Code": 0,
  "Result": {
    "PersistentVolumeClaimId": 0
  }
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

1.5.2. DeletePersistentVolumeClaim

调用DeletePersistentVolumeClaim删除一个持久化存储卷声明

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	DeletePersistentVolumeClaim	系统规定参数。取值：DeletePersistentVolumeClaim。
AppId	Long	是	0	应用id
EnvId	Long	是	0	环境id
PersistentVolumeClaimName	String	是	xxx	持久化存储卷声明名称

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功
RequestId	String	xx	请求id
ErrMsg	String	xx	错误信息
Result	Struct		返回结果
Success	Boolean	true	是否成功

示例

请求示例

```
http(s)://[Endpoint]/?Action=DeletePersistentVolumeClaim
&AppId=0
&EnvId=0
&PersistentVolumeClaimName=xxx
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<DeletePersistentVolumeClaimResponse>
      <RequestId>xx-62FB-4409-B718-A0CEF39B8F57</RequestId>
      <Code>0</Code>
      <Result>
            <Success>true</Success>
      </Result>
</DeletePersistentVolumeClaimResponse>
```

`JSON` 格式

```
{
  "RequestId": "xx-62FB-4409-B718-A0CEF39B8F57",
  "Code": 0,
  "Result": {
    "Success": true
  }
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

1.5.3. ListPersistentVolumeClaim

调用ListPersistentVolumeClaim查询持久化存储卷声明列表

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	ListPersistentVolumeClaim	系统规定参数。取值：ListPersistentVolumeClaim。
AppId	Long	是	0	应用id
EnvId	Long	是	0	环境id
PageNumber	Integer	是	1	页数
PageSize	Integer	是	10	每页大小

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功
PageNumber	Integer	1	页数
RequestId	String	xx	请求id
PageSize	Integer	10	每页大小
TotalCount	Long	5	总数
ErrorMsg	String	xx	错误信息
Data	Array		返回的数据
Name	String	xx	持久化存储卷声明的名称
Capacity	String	500Gi	容量
AccessModes	String	ReadWriteMany	访问模式
Status	String	Bound	状态
StorageClass	String	xx	存储类
VolumeName	String	xx	绑定到的持久化存储卷的名称
CreateTime	String	2020-01-15 12:00:00	创建时间

示例

请求示例

```
http(s)://[Endpoint]/?Action=ListPersistentVolumeClaim
&AppId=0
&EnvId=0
&PageNumber=1
&PageSize=10
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<ListPersistentVolumeClaimResponse>
      <TotalCount>2</TotalCount>
      <PageSize>10</PageSize>
      <RequestId>xx-DF68-4F22-921C-9DAB0BFBA777</RequestId>
      <PageNumber>1</PageNumber>
      <Data>
            <Status>Bound</Status>
            <VolumeName>console-create-pv</VolumeName>
            <Capacity>300Gi</Capacity>
            <StorageClass>nas</StorageClass>
            <CreateTime>2019-12-20 12:59:35</CreateTime>
            <AccessModes>ReadWriteMany</AccessModes>
            <Name>pvc-test</Name>
      </Data>
      <Data>
            <Status>Bound</Status>
            <VolumeName>jck-pv-test</VolumeName>
            <Capacity>200Gi</Capacity>
            <StorageClass>test-sc</StorageClass>
            <CreateTime>2019-12-20 13:22:00</CreateTime>
            <AccessModes>ReadWriteMany</AccessModes>
            <Name>jck-pvc-test</Name>
      </Data>
      <Code>0</Code>
</ListPersistentVolumeClaimResponse>
```

`JSON` 格式

```
{
  "TotalCount": 2,
  "PageSize": 10,
  "RequestId": "xx-DF68-4F22-921C-9DAB0BFBA777",
  "PageNumber": 1,
  "Data": [
    {
      "Status": "Bound",
      "VolumeName": "console-create-pv",
      "Capacity": "300Gi",
      "StorageClass": "nas",
      "CreateTime": "2019-12-20 12:59:35",
      "AccessModes": "ReadWriteMany",
      "Name": "pvc-test"
    },
    {
      "Status": "Bound",
      "VolumeName": "jck-pv-test",
      "Capacity": "200Gi",
      "StorageClass": "test-sc",
      "CreateTime": "2019-12-20 13:22:00",
      "AccessModes": "ReadWriteMany",
      "Name": "jck-pvc-test"
    }
  ],
  "Code": 0
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

1.6. 应用分组管理

1.6.1. CreateAppGroup

调用CreateAppGroup创建一个应用分组

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	CreateAppGroup	系统规定参数。取值：CreateAppGroup。

BizCode	String	是	JST	业务域限制 JST：聚石塔。 NEW_RETAIL：线下零售。 MINI_APP：轻应用云。 SUPPLY：供应链。 MESSAGE：消息业务。
Name	String	是	test-group	分组名称

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功
ErrMsg	String	xx	错误信息
RequestId	String	C181375C-xxx-xxxx-8D9B-4F723D3FFF1E	请求id
Result	Struct		返回结果
AppGroupId	Long	0	分组id

示例

请求示例

```
http(s)://[Endpoint]/?Action=CreateAppGroup
&BizCode=JST
&Name=test-group
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<CreateAppGroupResponse>
      <RequestId>C181375C-xxx-xxxx-8D9B-4F723D3FFF1E</RequestId>
      <ErrMsg>xx</ErrMsg>
      <Code>0</Code>
      <Result>
            <AppGroupId>0</AppGroupId>
      </Result>
</CreateAppGroupResponse>
```

`JSON` 格式

```
{"RequestId":"C181375C-xxx-xxxx-8D9B-4F723D3FFF1E","ErrMsg":"xx","Code":"0","Result":{"
AppGroupId":"0"}}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

1.6.2. ListAppGroup

调用ListAppGroup查询用户的分组列表

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	ListAppGroup	系统规定参数。取值：ListAppGroup。
BizCode	String	是	JST	业务域限制 JST：聚石塔。 NEW_RETAIL：线下零售。 MINI_APP：轻应用云。 SUPPLY：供应链。 MESSAGE：消息业务。
PageNumber	Integer	是	1	页数
PageSize	Integer	是	10	每页大小

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功
Data	Array of AppGroupDetail		返回数据
BizName	String	JST	业务域名称
Id	Long	0	分组id
Name	String	test-group	分组名称
ErrorMsg	String	xx	错误信息

PageNumber	Integer	1	页数
PageSize	Integer	10	每页大小
RequestId	String	C181375C-xxx-xxxx-8D9B-4F723D3FFF1E	请求id
TotalCount	Long	23	总数

示例

请求示例

```
http(s)://[Endpoint]/?Action=ListAppGroup
&BizCode=JST
&PageNumber=1
&PageSize=10
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<ListAppGroupResponse>
      <TotalCount>23</TotalCount>
      <PageSize>10</PageSize>
      <RequestId>C181375C-xxx-xxxx-8D9B-4F723D3FFF1E</RequestId>
      <PageNumber>1</PageNumber>
      <ErrorMsg>xx</ErrorMsg>
      <Data>
            <BizName>JST</BizName>
            <Id>0</Id>
            <Name>test-group</Name>
      </Data>
      <Code>0</Code>
</ListAppGroupResponse>
```

`JSON` 格式

```
{"TotalCount":"23","PageSize":"10","RequestId":"C181375C-xxx-xxxx-8D9B-
4F723D3FFF1E","PageNumber":"1","ErrorMsg":"xx","Data":
[{"BizName":"JST","Id":"0","Name":"test-group"}],"Code":"0"}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

1.6.3. BindGroup

将一个应用加入到某个分组中。

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	BindGroup	系统规定参数。取值：BindGroup。
AppId	Long	是	0	应用id
BizCode	String	是	JST	业务域限制 JST：聚石塔。 NEW_RETAIL：线下零售。 MINI_APP：轻应用云。 SUPPLY：供应链。 MESSAGE：消息业务。
Name	String	是	test-group	要绑定的分组名称

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功
ErrMsg	String	xx	错误信息
RequestId	String	C181375C-xxx-xxxx-8D9B-4F723D3FFF1E	请求id
Result	Struct		返回结果
Success	Boolean	true	是否绑定成功

示例

请求示例

```
http(s)://[Endpoint]/?Action=BindGroup
&AppId=0
&BizCode=JST
&Name=test-group
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<BindGroupResponse>
      <RequestId>C181375C-xxx-xxxx-8D9B-4F723D3FFF1E</RequestId>
      <ErrMsg>xx</ErrMsg>
      <Code>0</Code>
      <Result>
            <Success>true</Success>
      </Result>
</BindGroupResponse>
```

`JSON` 格式

```
{"RequestId":"C181375C-xxx-xxxx-8D9B-4F723D3FFF1E","ErrMsg":"xx","Code":"0","Result":{"
Success":"true"}}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

1.6.4. UnbindGroup

调用UnbindGroup，从分组中解绑应用

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	UnbindGroup	系统规定参数。取值：UnbindGroup。
AppId	Long	是	0	应用id
BizCode	String	是	JST	业务域限制 JST：聚石塔。 NEW_RETAIL：线下零售。 MINI_APP：轻应用云。 SUPPLY：供应链。 MESSAGE：消息业务。
Name	String	是	test-group	分组名称

返回数据

名称	类型	示例值	描述

Code	Integer	0	返回码，0表示成功
ErrMsg	String	xx	错误信息
RequestId	String	C181375C-xxx-xxxx-8D9B-4F723D3FFF1E	请求id
Result	Struct		返回结果
Success	Boolean	true	解绑是否成功

示例

请求示例

```
http(s)://[Endpoint]/?Action=UnbindGroup
&AppId=0
&BizCode=JST
&Name=test-group
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<UnbindGroupResponse>
      <RequestId>C181375C-xxx-xxxx-8D9B-4F723D3FFF1E</RequestId>
      <ErrMsg>xx</ErrMsg>
      <Code>0</Code>
      <Result>
            <Success>true</Success>
      </Result>
</UnbindGroupResponse>
```

`JSON` 格式

```
{"RequestId":"C181375C-xxx-xxxx-8D9B-4F723D3FFF1E","ErrMsg":"xx","Code":"0","Result":{"
Success":"true"}}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

1.6.5. ListAppGroupMapping

调用ListAppGroupMapping查询分组和应用的映射列表，即查询分组下的应用

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	ListAppGroupMapping	系统规定参数。取值：ListAppGroupMapping。
BizCode	String	是	JST	业务域限制 JST：聚石塔。 NEW_RETAIL：线下零售。 MINI_APP：轻应用云。 SUPPLY：供应链。 MESSAGE：消息业务。
Name	String	是	test-group	分组名称
PageNumber	Integer	是	1	页数
PageSize	Integer	是	10	每页大小

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功
Data	Array of AppGroupMappingDetail		返回数据
AppId	Long	0	应用id
GroupId	Long	0	分组id
Id	Long	0	映射记录的id
Name	String	test-group	分组名称
ErrorMsg	String	xx	错误信息
PageNumber	Integer	1	页数
PageSize	Integer	10	每页大小

RequestId	String	C181375C-xxx-xxxx-8D9B-4F723D3FFF1E	请求id
TotalCount	Long	20	总数

示例

请求示例

```
http(s)://[Endpoint]/?Action=ListAppGroupMapping
&BizCode=JST
&Name=test-group
&PageNumber=1
&PageSize=10
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<ListAppGroupMappingResponse>
      <TotalCount>20</TotalCount>
      <PageSize>10</PageSize>
      <RequestId>C181375C-xxx-xxxx-8D9B-4F723D3FFF1E</RequestId>
      <PageNumber>1</PageNumber>
      <ErrorMsg>xx</ErrorMsg>
      <Data>
            <AppId>0</AppId>
            <Id>0</Id>
            <GroupId>0</GroupId>
            <Name>test-group</Name>
      </Data>
      <Code>0</Code>
</ListAppGroupMappingResponse>
```

`JSON` 格式

```
{"TotalCount":"20","PageSize":"10","RequestId":"C181375C-xxx-xxxx-8D9B-
4F723D3FFF1E","PageNumber":"1","ErrorMsg":"xx","Data":
[{"AppId":"0","Id":"0","GroupId":"0","Name":"test-group"}],"Code":"0"}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

1.6.6. DeleteAppGroup

调用DeleteAppGroup删除一个应用分组

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	DeleteAppGroup	系统规定参数。取值：DeleteAppGroup。
Force	Boolean	是	false	是否强制删除，true: 删除分组下的所有和应用的关联关系，false: 存在关联关系无法删除
GroupId	Long	是	0	分组id

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功
ErrMsg	String	xx	错误信息
RequestId	String	C181375C-xxx-xxxx-8D9B-4F723D3FFF1E	请求id
Result	Struct		返回结果
Success	Boolean	true	是否删除成功

示例

请求示例

```
http(s)://[Endpoint]/?Action=DeleteAppGroup
&Force=false
&GroupId=0
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<DeleteAppGroupResponse>
      <RequestId>C181375C-xxx-xxxx-8D9B-4F723D3FFF1E</RequestId>
      <ErrMsg>xx</ErrMsg>
      <Code>0</Code>
      <Result>
            <Success>true</Success>
      </Result>
</DeleteAppGroupResponse>
```

`JSON` 格式

```
{"RequestId":"C181375C-xxx-xxxx-8D9B-4F723D3FFF1E","ErrMsg":"xx","Code":"0","Result":{"
Success":"true"}}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

2.用户管理

2.1. ListUsers

列举当前主账号下所有用户。

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	ListUsers	系统规定参数。取值：ListUsers。
PageNumber	Integer	是	1	页码
PageSize	Integer	是	10	页大小。

返回数据

名称	类型	示例值	描述
Code	Integer	200	返回的错误码
ErrorMsg	String	NPE	错误信息
PageNumber	Integer	1	当前页码
PageSize	Integer	10	页大小
RequestId	String	abcd	请求ID
TotalCount	Long	6	总记录数
Data	Array		返回数据
UserId	String	1234	用户ID
UserType	String	TAOBAO	用户类型。当前有两种类型：TAOBAO, DING_TALK

RealName	String	张三	真实名称

示例

请求示例

```
http(s)://[Endpoint]/?Action=ListUsers
&PageNumber=1
&PageSize=10
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<ListUserResponse>
          <PageNumber>1</PageNumber>
          <Data>
                <UserType>TAOBAO</UserType>
                <UserId>12345</UserId>
                <RealName>张三</RealName>
          </Data>
          <Data>
                <UserType>TAOBAO</UserType>
                <UserId>56789</UserId>
                <RealName>李四</RealName>
          </Data>
          <Data>
                <UserType>DINGDING_TALK</UserType>
                <UserId>abcdkfasf</UserId>
                <RealName>王五</RealName>
          </Data>
          <TotalCount>3</TotalCount>
          <PageSize>10</PageSize>
          <RequestId>1472B82E-C7A6-4A96-ADF6-3299BA1B4D44</RequestId>
          <Code>0</Code>
</ListUserResponse>
```

`JSON` 格式

```
{
  "PageNumber": 1,
  "Data": [
    {
      "UserType": "TAOBAO",
      "UserId": "12345",
      "RealName": "张三"
    },
    {
      "UserType": "TAOBAO",
      "UserId": "56789",
      "RealName": "李四"
    },
    {
      "UserType": "DINGDING_TALK",
      "UserId": "abcdkfasf",
      "RealName": "王五"
    }
  ],
  "TotalCount": 3,
  "PageSize": 10,
  "RequestId": "1472B82E-C7A6-4A96-ADF6-3299BA1B4D44",
  "Code": 0
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

3.应用运维

3.1. SLB接入

3.1.1. CreateSlbAP

调用CreateSlbAP创建一个SLB接入。通过SLB接入将部署在容器服务内的服务暴露并对外提供服务。

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	CreateSlbAP	系统规定参数。取值：CreateSlbAP。
EnvId	Long	是	12	环境id。
Name	String	是	xxx	SLB接入名称。关于SLB接入的一个简单描述，便于识别这个SLB接入的作用。
SlbId	String	是	lb-xxxxx	SLB的id。
ListenerPort	Integer	是	8080	监听的端口。也是负载均衡对外提供服务的端口，通常HTTP协议使用80端口，HTTPS协议使用443端口
RealServerPort	Integer	是	8080	后端服务在容器内真正监听的端口。
Protocol	String	是	TCP	协议。枚举值： TCP、UDP、HTTP、HTTPS
EstablishedTimeout	Integer	否	900	协议为TCP时指定。配置TCP连接超时，连接空闲时间超过该时长后，负载均衡会主动断开该连接，超时时间输入范围为10-900秒。
StickySession	Integer	否	1	是否开启会话保持。协议为HTTP或HTTPS时指定，枚举值： 0：不开启； 1：开启；

SslCertId	String	否	ssl-xxx	阿里云-【SSL证书】产品维护的证书id。
CookieTimeout	Integer	否	86400	会话保持超时时间。协议为HTTP或HTTPS时指定，输入范围为1-86400秒。

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功。
ErrMsg	String	this is a tip message	错误消息。
RequestId	String	FD0E1170-xxxx-xxxx-8D2F-0E32F904F169	请求id。
Success	Boolean	true	是否成功。
Result	Struct		结果。
SlbAPId	Long	111	SLB接入id。

示例

请求示例

```
http(s)://[Endpoint]/?Action=CreateSlbAP
&EnvId=12
&Name=xxx
&SlbId=lb-xxxxx
&ListenerPort=8080
&RealServerPort=8080
&Protocol=TCP
&EstablishedTimeout=900
&StickySession=1
&SslCertId=ssl-xxx
&CookieTimeout=86400
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<CreateSlbAPResponse>
      <RequestId>99FEB288-DFF3-4869-997A-3F2DCE2D641D</RequestId>
      <Code>0</Code>
      <Success>true</Success>
      <Result>
            <SlbAPId>123</SlbAPId>
      </Result>
</CreateSlbAPResponse>
```

`JSON` 格式

```
{
  "RequestId": "99FEB288-DFF3-4869-997A-3F2DCE2D641D",
  "Code": 0,
  "Success": true,
  "Result": {
    "SlbAPId": 123
  }
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

3.1.2. ModifySlbAP

调用ModifySlbAP修改SLB接入信息。只能修改协议的附加参数，不可修改协议。

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	ModifySlbAP	系统规定参数。取值：ModifySlbAP。
SlbAPId	Long	是	123	SLB接入的id。（非阿里云SLB的id）
Name	String	否	xxxx	SLB接入名称。关于SLB接入的一个简单描述，便于识别这个SLB接入的作用。
RealServerPort	Integer	否	8080	后端服务在容器内真正监听的端口。

EstablishedTimeout	Integer	否	900	协议为TCP时指定。配置TCP连接超时，连接空闲时间超过该时长后，负载均衡会主动断开该连接，超时时间输入范围为10-900秒。
StickySession	Integer	否	1	是否开启会话保持。协议为HTTP或HTTPS时指定，枚举值： 0：不开启； 1：开启；
SslCertId	String	否	ssl-xxxx	阿里云-【SSL证书】产品维护的证书id。
CookieTimeout	Integer	否	86400	会话保持超时时间。协议为HTTP或HTTPS时指定，输入范围为1-86400秒。

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功。
ErrMsg	String	this is a tip message	错误消息。
RequestId	String	FD0E1170-xxxx-xxxx-8D2F-0E32F904F169	请求id。
Success	Boolean	true	是否成功。

示例

请求示例

```
http(s)://[Endpoint]/?Action=ModifySlbAP
&SlbAPId=123
&Name=xxxx
&RealServerPort=8080
&EstablishedTimeout=900
&StickySession=1
&SslCertId=ssl-xxxx
&CookieTimeout=86400
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<ModifySlbAPResponse>
      <RequestId>A8D349A5-22C7-400E-9267-026CADE86C00</RequestId>
      <Code>0</Code>
      <Success>true</Success>
</ModifySlbAPResponse>
```

`JSON` 格式

```
{
  "RequestId": "A8D349A5-22C7-400E-9267-026CADE86C00",
  "Code": 0,
  "Success": true
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

3.1.3. DeleteSlbAP

调用DeleteSlbAP删除一个SLB接入。

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	DeleteSlbAP	系统规定参数。取值：DeleteSlbAP。
SlbAPId	Long	是	123	SLB接入id。

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功。
ErrMsg	String	this is a tip message	错误信息。
RequestId	String	FD0E1170-xxxx-xxxx-8D2F-0E32F904F169	请求id。
Success	Boolean	true	是否成功。

示例

请求示例

```
http(s)://[Endpoint]/?Action=DeleteSlbAP
&SlbAPId=123
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<DeleteSlbAPResponse>
      <RequestId>A8D349A5-22C7-400E-9267-026CADE86C00</RequestId>
      <Code>0</Code>
      <Success>true</Success>
</DeleteSlbAPResponse>
```

`JSON` 格式

```
{
  "RequestId": "A8D349A5-22C7-400E-9267-026CADE86C00",
  "Code": 0,
  "Success": true
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

3.1.4. DescribeSlbAPDetail

调用DescribeSlbAPDetail查询SLB接入的详细信息。

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	DescribeSlbAPDetail	系统规定参数。取值：DescribeSlbAPDetail。
SlbAPId	Long	是	123	SLB接入id。

返回数据

名称	类型	示例值	描述

Code	Integer	0	返回码，0表示成功。
ErrMsg	String	this is a tip message	错误信息。
RequestId	String	FD0E1170-xxxx-xxxx-8D2F-0E32F904F169	请求id。
Success	Boolean	true	是否成功。
Result	Struct		结果。
AppId	Long	123	应用id。
CookieTimeout	Integer	86400	会话保持超时时间。协议为HTTP或HTTPS时指定，输入范围为1-86400秒。
EnvId	Long	123	环境id。
EstablishedTimeout	Integer	900	协议为TCP时指定。配置TCP连接超时，连接空闲时间超过该时长后，负载均衡会主动断开该连接，超时时间输入范围为10-900秒。
ListenerPort	Integer	8080	监听的端口。也是负载均衡对外提供服务的端口，通常HTTP协议使用80端口，HTTPS协议使用443端口
Name	String	xxxx	SLB接入名称。关于SLB接入的一个简单描述，便于识别这个SLB接入的作用。
NetworkMode	String	internet	网络类型。枚举值： internet：外网 intranet：内网
Protocol	String	TCP	协议。枚举值： TCP、UDP、HTTP、HTTPS
RealServerPort	Integer	8080	后端服务在容器内真正监听的端口。
SlbAPId	Long	123	SLB接入id。
SlbId	String	lb-xxxxx	SLB（负载均衡）id。

SlbIp	String	127.0.0.1	SLB（负载均衡）ip。
SslCertId	String	ssl-xxxx	阿里云-【SSL证书】产品维护的证书id。
StickySession	Integer	1	是否开启会话保持。协议为HTTP或HTTPS时指定，枚举值： 0：不开启； 1：开启；

示例

请求示例

```
http(s)://[Endpoint]/?Action=DescribeSlbAPDetail
&SlbAPId=123
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<DescribeSlbAPDetailResponse>
      <RequestId>B3F7F240-41C0-47A1-9E44-916F174F6AF5</RequestId>
      <Code>0</Code>
      <Success>true</Success>
      <Result>
            <SlbId>lb-xxx</SlbId>
            <ListenerPort>8080</ListenerPort>
            <RealServerPort>8080</RealServerPort>
            <SlbAPId>123</SlbAPId>
            <EnvId>123</EnvId>
            <NetworkMode>intranet</NetworkMode>
            <Name>inner_xxx</Name>
            <SlbIp>192.168.49.51</SlbIp>
            <AppId>123</AppId>
            <Protocol>TCP</Protocol>
            <EstablishedTimeout>900</EstablishedTimeout>
      </Result>
</DescribeSlbAPDetailResponse>
```

`JSON` 格式

```
{
  "RequestId": "B3F7F240-41C0-47A1-9E44-916F174F6AF5",
  "Code": 0,
  "Success": true,
  "Result": {
    "SlbId": "lb-xxx",
    "ListenerPort": 8080,
    "RealServerPort": 8080,
    "SlbAPId": 123,
    "EnvId": 123,
    "NetworkMode": "intranet",
    "Name": "inner_xxx",
    "SlbIp": "192.168.49.51",
    "AppId": 123,
    "Protocol": "TCP",
    "EstablishedTimeout": 900
  }
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

3.1.5. ListSlbAPs

调用ListSlbAPs查询SLB接入列表。根据用户指定的查询条件分页查询SLB接入列表。

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	ListSlbAPs	系统规定参数。取值：ListSlbAPs。
AppId	Long	是	123	应用id。
PageNumber	Integer	是	1	页码。
PageSize	Integer	是	10	分页大小，每页的记录数量。
EnvId	Long	否	123	环境id。
Name	String	否	xxxx	名称。

NetworkMode	String	否	internet	网络类型。枚举值： internet：外网 intranet：内网
ProtocolList.N	RepeatList	否	TCP	协议。枚举值： TCP、UDP、HTTP、HTTPS
SlbId	String	否	111	SLB接入id。

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功。
ErrorMsg	String	this is a tip message	错误信息。
PageNumber	Integer	1	页码。
PageSize	Integer	10	分页大小。每页的记录数量。
RequestId	String	FD0E1170-xxxx-xxxx-8D2F-0E32F904F169	请求id。
TotalCount	Long	20	记录总数量。
Data	Array		分页数据。
AppId	Long	123	应用id。
CookieTimeout	Integer	86400	会话保持超时时间。协议为HTTP或HTTPS时指定，输入范围为1-86400秒。
EnvId	Long	123	环境id。
EstablishedTimeout	Integer	900	协议为TCP时指定。配置TCP连接超时，连接空闲时间超过该时长后，负载均衡会主动断开该连接，超时时间输入范围为10-900秒。

ListenerPort	Integer	8080	监听的端口。也是负载均衡对外提供服务的端口，通常HTTP协议使用80端口，HTTPS协议使用443端口
Name	String	xxxx	SLB接入名称。关于SLB接入的一个简单描述，便于识别这个SLB接入的作用。
NetworkMode	String	internet	网络类型。枚举值： internet：外网 intranet：内网
Protocol	String	TCP	协议。枚举值： TCP、UDP、HTTP、HTTPS
RealServerPort	Integer	8080	后端服务在容器内真正监听的端口。
SlbAPId	Long	123	SLB接入id。
SlbId	String	lb-xxxxx	SLB（负载均衡）id。
SlbIp	String	127.1.1.1	SLB（负载均衡）ip。
SslCertId	String	ssl-xxxx	阿里云-【SSL证书】产品维护的证书id。
StickySession	Integer	1	是否开启会话保持。协议为HTTP或HTTPS时指定，枚举值： 0：不开启； 1：开启；

示例

请求示例

```
http(s)://[Endpoint]/?Action=ListSlbAPs
&AppId=123
&EnvId=123
&Name=xxxx
&NetworkMode=internet
&PageNumber=1
&PageSize=10
&SlbId=111
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<ListSlbAPsResponse>
  <TotalCount>1</TotalCount>
  <PageSize>20</PageSize>
  <RequestId>E6E78956-B905-48FB-A1FF-28A0E8F40405</RequestId>
  <PageNumber>1</PageNumber>
  <Data>
        <SlbId>lb-k2jl80ttj26mzfjnw2cc2</SlbId>
        <ListenerPort>8080</ListenerPort>
        <RealServerPort>8080</RealServerPort>
        <AppId>123</AppId>
        <SlbIp>192.168.49.51</SlbIp>
        <SlbAPId>123</SlbAPId>
        <EnvId>123</EnvId>
        <NetworkMode>intranet</NetworkMode>
        <Protocol>TCP</Protocol>
        <EstablishedTimeout>900</EstablishedTimeout>
        <Name>xxxx</Name>
  </Data>
  <Code>0</Code>
</ListSlbAPsResponse>
```

`JSON` 格式

```
{
  "TotalCount": 1,
  "PageSize": 20,
  "RequestId": "E6E78956-B905-48FB-A1FF-28A0E8F40405",
  "PageNumber": 1,
  "Data": [
    {
      "SlbId": "lb-k2jl80ttj26mzfjnw2cc2",
      "ListenerPort": 8080,
      "RealServerPort": 8080,
      "AppId": 123,
      "SlbIp": "192.168.49.51",
      "SlbAPId": 123,
      "EnvId": 123,
      "NetworkMode": "intranet",
      "Protocol": "TCP",
      "EstablishedTimeout": 900,
      "Name": "xxxx"
    }
  ],
  "Code": 0
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

3.2. Service管理

3.2.1. ModifyService

调用ModifyService修改k8s service。

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	ModifyService	系统规定参数。取值：ModifyService。
ServiceId	Long	是	123	service id。
Name	String	否	xxxxx	service名称。关于service的一个简单描述，便于识别这个service的作用。
PortMappings.N.Protocol	String	否	TCP	协议。枚举值： TCP、UDP
PortMappings.N.TargetPort	String	否	8080	服务实际监听的端口。
PortMappings.N.Port	Integer	否	8080	service对外提供服务的端口。
PortMappings.N.Name	String	否	name1	port的名称。
PortMappings.N.NodePort	Integer	否	8080	k8s Node上暴露的端口。当ServiceType为NodePort时可指定。

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功。
ErrMsg	String	this is a tip message	错误信息。
RequestId	String	FD0E1170-xxxx-xxxx-8D2F-0E32F904F169	请求id。

Success	Boolean	true	是否成功。

示例

请求示例

```
http(s)://[Endpoint]/?Action=ModifyService
&Name=xxxxx
&ServiceId=123
&PortMappings.1.Protocol=TCP
&PortMappings.1.TargetPort=8080
&PortMappings.1.Port=8080
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<ModifyServiceResponse>
      <RequestId>4E56E0DD-173B-4872-A7E4-A5DD1A60B29E</RequestId>
      <Code>0</Code>
      <Success>true</Success>
</ModifyServiceResponse>
```

`JSON` 格式

```
{
  "RequestId": "4E56E0DD-173B-4872-A7E4-A5DD1A60B29E",
  "Code": 0,
  "Success": true
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

3.2.2. DeleteService

调用DeleteService删除一个service。

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	DeleteService	系统规定参数。取值：DeleteService。

ServiceId	Long	是	123	service id。

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功。
ErrMsg	String	this is a tip message	错误信息。
RequestId	String	FD0E1170-xxxx-xxxx-8D2F-0E32F904F169	请求id。
Success	Boolean	true	是否成功。

示例

请求示例

```
http(s)://[Endpoint]/?Action=DeleteService
&ServiceId=123
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<DeleteServiceResponse>
      <RequestId>4E56E0DD-173B-4872-A7E4-A5DD1A60B29E</RequestId>
      <Code>0</Code>
      <Success>true</Success>
</DeleteServiceResponse>
```

`JSON` 格式

```
{
  "RequestId": "4E56E0DD-173B-4872-A7E4-A5DD1A60B29E",
  "Code": 0,
  "Success": true
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

3.2.3. ListServices

调用ListServices查询Service列表。根据用户指定的查询条件分页查询Service列表。

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	ListServices	系统规定参数。取值：ListServices。
AppId	Long	是	123	应用id。
PageNumber	Integer	是	1	页码。
PageSize	Integer	是	10	分页大小，每页的记录数量。
EnvId	Long	否	123	环境id。
Name	String	否	xxx	service名称。关于service的一个简单描述，便于识别这个service的作用。
ServiceType	String	否	ClusterIP	service的类型。枚举值： ClusterIP：虚拟集群IP NodePort：节点端口

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功。
ErrorMsg	String	this is a tip message	错误信息。
PageNumber	Integer	1	页码。
PageSize	Integer	10	分页大小，每页的记录数量。
RequestId	String	FD0E1170-xxxx-xxxx-8D2F-0E32F904F169	请求id。
TotalCount	Long	20	记录总数量。

Data	Array		分页数据。
AppId	Long	123	应用id。
EnvId	Long	123	环境id。
Headless	Boolean	true	headless。当ServiceType为ClusterIP时可指定。
K8sServiceId	String	xxxx	对应k8s service name。
Name	String	xxx	service名称。关于service的一个简单描述，便于识别这个service的作用。
ServiceId	Long	123	service id。
ServiceType	String	ClusterIP	service的类型。枚举值： ClusterIP：虚拟集群IP NodePort：节点端口
PortMappings	Array		端口映射。
Name	String	xxx	port的名称。
NodePort	Integer	8080	k8s Node上暴露的端口。当ServiceType为NodePort时可指定。
Port	Integer	8080	service对外提供服务的端口。
Protocol	String	TCP	协议。枚举值： TCP、UDP
TargetPort	String	8080	服务实际监听的端口。

示例

请求示例

```
http(s)://[Endpoint]/?Action=ListServices
&AppId=123
&EnvId=123
&Name=xxx
&PageNumber=1
&PageSize=10
&ServiceType=ClusterIP
&<公共请求参数>
```

正常返回示例

`XML` **格式**

```
<ListServicesResponse>
      <TotalCount>2</TotalCount>
      <PageSize>10</PageSize>
      <RequestId>7E191918-E590-4665-B260-E9267EFDC286</RequestId>
      <PageNumber>1</PageNumber>
      <Data>
            <PortMappings>
                  <TargetPort>8080</TargetPort>
                  <Port>8080</Port>
                  <Protocol>TCP</Protocol>
                  <Name>test1</Name>
            </PortMappings>
            <PortMappings>
                  <TargetPort>8081</TargetPort>
                  <Port>8081</Port>
                  <Protocol>TCP</Protocol>
                  <Name>test2</Name>
            </PortMappings>
            <AppId>123</AppId>
            <K8sServiceId>lingfeng-service2</K8sServiceId>
            <ServiceType>ClusterIP</ServiceType>
            <EnvId>123</EnvId>
            <Headless>true</Headless>
            <Name>API创建测试k8sService</Name>
            <ServiceId>123</ServiceId>
      </Data>
      <Code>0</Code>
</ListServicesResponse>
```

`JSON` **格式**

```
{
  "TotalCount": 2,
  "PageSize": 10,
  "RequestId": "7E191918-E590-4665-B260-E9267EFDC286",
  "PageNumber": 1,
  "Data": [
    {
      "PortMappings": [
        {
          "TargetPort": "8080",
          "Port": 8080,
          "Protocol": "TCP",
          "Name": "test1"
        },
        {
          "TargetPort": "8081",
          "Port": 8081,
          "Protocol": "TCP",
          "Name": "test2"
        }
      ],
      "AppId": 123,
      "K8sServiceId": "lingfeng-service2",
      "ServiceType": "ClusterIP",
      "EnvId": 123,
      "Headless": true,
      "Name": "API创建测试k8sService",
      "ServiceId": 123
    }
  ],
  "Code": 0
}
```


错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

3.2.4. DescribeServiceDetail

调用DescribeServiceDetail查询service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。

请求参数

名称	类型	是否必选	示例值	描述

Action	String	是	DescribeServiceDetail	系统规定参数。取值：DescribeServiceDetail。
ServiceId	Long	是	123	service id

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功。
ErrMsg	String	this is a tip message	错误信息。
RequestId	String	FD0E1170-xxxx-xxxx-8D2F-0E32F904F169	请求id。
Success	Boolean	true	是否成功。
Result	Struct		结果。
AppId	Long	123	应用id。
EnvId	Long	123	环境id。
Headless	Boolean	true	headless。当ServiceType为ClusterIP时可指定。
K8sServiceId	String	xxxxx	对应k8s service name。
Name	String	xxx	service名称。关于service的一个简单描述，便于识别这个service的作用。
ServiceId	Long	123	service id。
ServiceType	String	ClusterIP	service的类型。枚举值： ClusterIP：虚拟集群IP NodePort：节点端口
PortMappings	Array		端口映射。

Name	String	xxxx	port的名称。
NodePort	Integer	8080	k8s Node上暴露的端口。当ServiceType为NodePort时可指定。
Port	Integer	8080	service对外提供服务的端口。
Protocol	String	TCP	协议。枚举值： TCP、UDP
TargetPort	String	8080	服务实际监听的端口。

示例

请求示例

```
http(s)://[Endpoint]/?Action=DescribeServiceDetail
&ServiceId=123
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<DescribeServiceDetailResponse>
      <RequestId>0CC86F48-11ED-4432-AA56-4BFD49E963F2</RequestId>
      <Code>0</Code>
      <Success>true</Success>
      <Result>
            <PortMappings>
                  <TargetPort>8080</TargetPort>
                  <Port>8080</Port>
                  <Protocol>TCP</Protocol>
                  <Name>test1</Name>
            </PortMappings>
            <PortMappings>
                  <TargetPort>8081</TargetPort>
                  <Port>8081</Port>
                  <Protocol>TCP</Protocol>
                  <Name>test2</Name>
            </PortMappings>
            <AppId>123</AppId>
            <K8sServiceId>lingfeng-service2</K8sServiceId>
            <ServiceType>ClusterIP</ServiceType>
            <EnvId>123</EnvId>
            <Headless>true</Headless>
            <Name>API创建测试k8sService</Name>
            <ServiceId>123</ServiceId>
      </Result>
</DescribeServiceDetailResponse>
```

`JSON` **格式**

```
{
  "RequestId": "0CC86F48-11ED-4432-AA56-4BFD49E963F2",
  "Code": 0,
  "Success": true,
  "Result": {
    "PortMappings": [
      {
        "TargetPort": "8080",
        "Port": 8080,
        "Protocol": "TCP",
        "Name": "test1"
      },
      {
        "TargetPort": "8081",
        "Port": 8081,
        "Protocol": "TCP",
        "Name": "test2"
      }
    ],
    "AppId": 123,
    "K8sServiceId": "lingfeng-service2",
    "ServiceType": "ClusterIP",
    "EnvId": 123,
    "Headless": true,
    "Name": "API创建测试k8sService",
    "ServiceId": 123
  }
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

3.2.5. CreateService

调用CreateService创建一个k8s service。通过k8s service将部署在容器服务内的服务暴露并对外提供服务。

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	CreateService	系统规定参数。取值：CreateService。
EnvId	Long	是	123	环境id。

Name	String	是	xxxx	service名称。关于service的一个简单描述，便于识别这个service的作用。
ServiceType	String	是	ClusterIP	service的类型。枚举值： ClusterIP：虚拟集群IP NodePort：节点端口
K8sServiceId	String	是	test	对应k8s service name。
PortMappings.N.Protocol	String	是	TCP	协议。枚举值： TCP、UDP
PortMappings.N.TargetPort	String	是	8080	服务实际监听的端口。
PortMappings.N.Port	Integer	是	8080	service对外提供服务的端口。
PortMappings.N.Name	String	是	name1	port的名称。
Headless	Boolean	否	true	headless。当ServiceType为ClusterIP时可指定。
PortMappings.N.NodePort	Integer	否	8080	k8s Node上暴露的端口。当ServiceType为NodePort时可指定。

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功。
ErrMsg	String	this is a tip message	错误信息。
RequestId	String	FD0E1170-xxxx-xxxx-8D2F-0E32F904F169	请求id。
Success	Boolean	true	是否成功。
Result	Struct		结果。
ServiceId	Long	123	service id

示例

请求示例

```
http(s)://[Endpoint]/?Action=CreateService
&EnvId=123
&Name=xxxx
&ServiceType=ClusterIP
&K8sServiceId=test
&Headless=true
&PortMappings.1.Protocol=TCP
&PortMappings.1.TargetPort=8080
&PortMappings.1.Port=8080
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<CreateServiceResponse>
      <RequestId>4E56E0DD-173B-4872-A7E4-A5DD1A60B29E</RequestId>
      <Code>0</Code>
      <Success>true</Success>
      <Result>
            <ServiceId>123</ServiceId>
      </Result>
</CreateServiceResponse>
```

`JSON` 格式

```
{
  "RequestId": "4E56E0DD-173B-4872-A7E4-A5DD1A60B29E",
  "Code": 0,
  "Success": true,
  "Result": {
    "ServiceId": 123
  }
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

3.3. 任务管理

3.3.1. ListJobHistories

调用ListJobHistories查询Job的执行历史

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	ListJobHistories	系统规定参数。取值：ListJobHistories。
AppId	Long	是	0	应用id
EnvId	Long	是	0	环境id
PageNumber	Integer	是	1	页数
PageSize	Integer	是	10	页大小
Status	String	是	SUCCESS	任务的状态，取值范围：SUCCESS、FAILED、ALL

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功
Data	Array		返回的数据
ActiveDeadlineSeconds	Integer	30	任务启动期限秒数
BackoffLimit	Integer	6	任务失败后重试的次数，默认6次
CompletionTime	String	2020-05-19 10:01:00	完成时间
Completions	Integer	1	任务期望的成功完成pod数
Failed	Integer	0	失败任务数
Message	String	xx	失败信息
Name	String	xx	任务名称

Parallelism	Integer	1	任务并行度
PodList	List	[xxx, xxx]	该job管理的pod名字列表
StartTime	String	2020-05-19 10:00:00	任务开始时间
Succeeded	Integer	1	成功的任务数
ErrorMsg	String	xx	错误信息
PageNumber	Integer	1	页数
PageSize	Integer	10	页大小
RequestId	String	xx	请求id
TotalCount	Long	5	总条数

示例

请求示例

```
http(s)://[Endpoint]/?Action=ListJobHistories
&AppId=0
&EnvId=0
&PageNumber=1
&PageSize=10
&Status=SUCCESS
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<ListJobHistoriesResponse>
      <TotalCount>5</TotalCount>
      <RequestId>xx</RequestId>
      <PageSize>10</PageSize>
      <PageNumber>1</PageNumber>
      <ErrorMsg>xx</ErrorMsg>
      <Data>
            <ActiveDeadlineSeconds>30</ActiveDeadlineSeconds>
            <Succeeded>1</Succeeded>
            <BackoffLimit>6</BackoffLimit>
            <Message>xx</Message>
            <CompletionTime>2020-05-19 10:01:00</CompletionTime>
            <Failed>0</Failed>
            <StartTime>2020-05-19 10:00:00</StartTime>
            <Parallelism>1</Parallelism>
            <Name>xx</Name>
            <Completions>1</Completions>
      </Data>
      <Data>
            <PodList>[xxx, xxx]</PodList>
      </Data>
      <Code>0</Code>
</ListJobHistoriesResponse>
```

`JSON` **格式**

```
{
  "TotalCount": "5",
  "RequestId": "xx",
  "PageSize": "10",
  "PageNumber": "1",
  "ErrorMsg": "xx",
  "Data": [
    {
      "ActiveDeadlineSeconds": "30",
      "Succeeded": "1",
      "BackoffLimit": "6",
      "Message": "xx",
      "CompletionTime": "2020-05-19 10:01:00",
      "Failed": "0",
      "StartTime": "2020-05-19 10:00:00",
      "Parallelism": "1",
      "Name": "xx",
      "Completions": "1"
    },
    {
      "PodList": "[xxx, xxx]"
    }
  ],
  "Code": "0"
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

3.3.2. DescribeJobLog

调用DescribeJobLog查询任务的执行日志

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	DescribeJobLog	系统规定参数。取值：DescribeJobLog。
AppId	Long	是	0	应用id
EnvId	Long	是	0	环境id
PodName	String	是	xxx	执行任务的pod名称，通过ListJobHistories接口里可以拿到

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功
ErrMsg	String	xx	错误信息
RequestId	String	xx	请求id
Result	Struct		返回结果
AppId	Long	0	应用id
EnvId	Long	0	环境id
Events	Array		执行任务的pod的事件列表

Action	String	xx	Action
Count	Integer	1	数量
EventTime	String	2020-01-05T18:38Z	事件发生时间
FirstTimestamp	String	2020-01-05T18:38Z	首次时间
LastTimestamp	String	2020-01-05T18:38Z	最后一次时间
Mesage	String	Successfully assigned	事件消息
Reason	String	Scheduled	事件原因
Type	String	Normal	事件类型
JobLog	String	xxx	执行任务的日志
PodName	String	xx	执行任务的pod名称

示例

请求示例

```
http(s)://[Endpoint]/?Action=DescribeJobLog
&AppId=0
&EnvId=0
&PodName=xxx
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<DescribeJobLogResponse>
      <RequestId>xx</RequestId>
      <ErrMsg>xx</ErrMsg>
      <Code>0</Code>
      <Result>
            <PodName>xx</PodName>
            <AppId>0</AppId>
            <Events>
                  <Action>xx</Action>
                  <Type>Normal</Type>
                  <LastTimestamp>2020-01-05T18:38Z</LastTimestamp>
                  <EventTime>2020-01-05T18:38Z</EventTime>
                  <Count>1</Count>
                  <FirstTimestamp>2020-01-05T18:38Z</FirstTimestamp>
                  <Mesage>Successfully assigned</Mesage>
                  <Reason>Scheduled</Reason>
            </Events>
            <JobLog>xxx</JobLog>
            <EnvId>0</EnvId>
      </Result>
</DescribeJobLogResponse>
```

`JSON` **格式**

```
{
  "RequestId": "xx",
  "ErrMsg": "xx",
  "Code": "0",
  "Result": {
    "PodName": "xx",
    "AppId": "0",
    "Events": [
      {
        "Action": "xx",
        "Type": "Normal",
        "LastTimestamp": "2020-01-05T18:38Z",
        "EventTime": "2020-01-05T18:38Z",
        "Count": "1",
        "FirstTimestamp": "2020-01-05T18:38Z",
        "Mesage": "Successfully assigned",
        "Reason": "Scheduled"
      }
    ],
    "JobLog": "xxx",
    "EnvId": "0"
  }
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

4.应用监控

4.1. DescribePodContainerLogList

调用DescribePodContainerLogList查询实例中的容器实时日志

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	DescribePodContainerLogList	系统规定参数。取值：DescribePodContainerLogList。
AppId	Long	是	1122	云应用id
EnvId	Long	是	1122	云应用环境id
Line	Integer	是	200	日志行数
PodName	String	是	jck-deployment-yacs-24209-25073-728587-5757b8df59-rzpw6	实例名称

返回数据

名称	类型	示例值	描述
Code	Integer	200	状态码
ErrMsg	String	xxx	错误信息
RequestId	String	xxx-xxx	请求id
Result	Struct		结果
ContainerLogList	Array of PodContainerLog		容器列表
ContainerName	String	jck-container-xxx	容器名称

Content	String	xxx	日志内容
PodName	String	xxx	实例名称
Success	Boolean	true	是否成功

示例

请求示例

```
http(s)://[Endpoint]/?Action=DescribePodContainerLogList
&AppId=1122
&EnvId=1122
&Line=200
&PodName=jck-deployment-yacs-24209-25073-728587-5757b8df59-rzpw6
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<DescribePodContainerLogList>
  <RequestId>F2053086-E9E0-1164-B05A-9CF5C79BDAAF</RequestId>
  <Code>0</Code>
  <Success>true</Success>
  <Result>
        <ContainerLogList>
              <PodName>jck-deployment-yacs-24209-25073-728587-5757b8df59-rzpw6</PodName>
              <ContainerName>jck-app-container-24209-25073</ContainerName>
              <Content>-server -Xms2g -Xmx2g -Xmn1g -XX:MetaspaceSize=256m -
XX:MaxMetaspaceSize=256m -Duser.timezone=Asia/Shanghai
--management.info.build.mode=full --spring.profiles.active=production --
logging.path=/acs/log --logging.file=/acs/log/application.log
8080
8081
war inf dir exist
Picked up JAVA_TOOL_OPTIONS:  -Doneagent.plugin.arms-agent.enabled=true -
javaagent:/home/admin/.opt/ArmsAgent/arms-bootstrap-1.7.0-SNAPSHOT.jar -
Darms.licenseKey=ekkf778j1t@d643cdcd427b3bf -Darms.appId=ekkf778j1t@e8bd65c86fc2c61 -Da
rms.agent.env=ACSK8S  -Darms.agent.podinfo.path=/etc/podinfo
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
2021-07-19 16:38:23 [WARN ](com.navercorp.pinpoint.bootstrap.PinpointBootStrap) arms ag
ent start cost: 4607ms

  .   ____          _            __ _ _
 /\\ / ___'_ __ _ _(_)_ __  __ _ \ \ \ \
( ( )\___ | '_ | '_| | '_ \/ _` | \ \ \ \
 \\/  ___)| |_)| | | | | || (_| |  ) ) ) )
  '  |____| .__|_| |_|_| |_\__, | / / / /
 =========|_|==============|___/=/_/_/_/
```

```
 :: Spring Boot ::        (v2.0.2.RELEASE)

2021-07-19 16:38:24.932  INFO 9 --- [           main] com.tmall.ews.Application
: Starting Application on jck-deployment-yacs-24209-25073-728587-5757b8df59-rzpw6 with
PID 9 (/acs/code/WEB-INF/classes started by root in /acs/code)
2021-07-19 16:38:24.933  INFO 9 --- [           main] com.tmall.ews.Application
: The following profiles are active: production
2021-07-19 16:38:25.005  INFO 9 --- [           main]
ConfigServletWebServerApplicationContext : Refreshing
org.springframework.boot.web.servlet.context.AnnotationConfigServletWebServerApplicationC
text@3d7cc3cb: startup date [Mon Jul 19 16:38:25 CST 2021]; root of context hierarchy
2021-07-19 16:38:26.706  INFO 9 --- [           main]
o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat initialized with port(s): 8080 (http)
2021-07-19 16:38:26.742  INFO 9 --- [           main]
o.apache.catalina.core.StandardService   : Starting service [Tomcat]
2021-07-19 16:38:26.742  INFO 9 --- [           main]
org.apache.catalina.core.StandardEngine  : Starting Servlet Engine: Apache
Tomcat/8.5.31
2021-07-19 16:38:26.754  INFO 9 --- [ost-startStop-1]
o.a.catalina.core.AprLifecycleListener   : The APR based Apache Tomcat Native library w
hich allows optimal performance in production environments was not found on the java.li
brary.path: [/usr/java/packages/lib/amd64:/usr/lib64:/lib64:/lib:/usr/lib]
2021-07-19 16:38:26.869  INFO 9 --- [ost-startStop-1] o.a.c.c.C.[Tomcat].[localhost].[/
]       : Initializing Spring embedded WebApplicationContext
2021-07-19 16:38:26.869  INFO 9 --- [ost-startStop-1] o.s.web.context.ContextLoader
: Root WebApplicationContext: initialization completed in 1864 ms
2021-07-19 16:38:27.194  INFO 9 --- [ost-startStop-1]
o.s.b.w.servlet.ServletRegistrationBean  : Servlet dispatcherServlet mapped to [/]
2021-07-19 16:38:27.272  INFO 9 --- [ost-startStop-1]
o.s.b.w.servlet.FilterRegistrationBean   : Mapping filter: 'characterEncodingFilter' to
: [/*]
2021-07-19 16:38:27.273  INFO 9 --- [ost-startStop-1]
o.s.b.w.servlet.FilterRegistrationBean   : Mapping filter: 'hiddenHttpMethodFilter' to:
[/*]
2021-07-19 16:38:27.273  INFO 9 --- [ost-startStop-1]
o.s.b.w.servlet.FilterRegistrationBean   : Mapping filter: 'httpPutFormContentFilter' t
o: [/*]
2021-07-19 16:38:27.273  INFO 9 --- [ost-startStop-1]
o.s.b.w.servlet.FilterRegistrationBean   : Mapping filter: 'requestContextFilter' to: [
/*]
2021-07-19 16:38:27.273  INFO 9 --- [ost-startStop-1]
o.s.b.w.servlet.FilterRegistrationBean   : Mapping filter: 'httpTraceFilter' to: [/*]
2021-07-19 16:38:27.274  INFO 9 --- [ost-startStop-1]
o.s.b.w.servlet.FilterRegistrationBean   : Mapping filter: 'webMvcMetricsFilter' to: [/
*]
2021-07-19 16:38:27.423  INFO 9 --- [           main]
o.s.w.s.handler.SimpleUrlHandlerMapping  : Mapped URL path [/**/favicon.ico] onto handl
er of type [class org.springframework.web.servlet.resource.ResourceHttpRequestHandler]
2021-07-19 16:38:27.653  INFO 9 --- [           main]
s.w.s.m.m.a.RequestMappingHandlerAdapter : Looking for @ControllerAdvice:
org.springframework.boot.web.servlet.context.AnnotationConfigServletWebServerApplicationC
text@3d7cc3cb: startup date [Mon Jul 19 16:38:25 CST 2021]; root of context hierarchy
2021-07-19 16:38:27.750  INFO 9 --- [           main]
s.w.s.m.m.a.RequestMappingHandlerMapping : Mapped "{[/__mse__/readiness]}" onto public
```

```
org.springframework.http.ResponseEntity
com.navercorp.pinpoint.plugin.micro.service.interceptor.Mse_Readiness_Controller.healthCh
k()
2021-07-19 16:38:27.752  INFO 9 --- [           main]
s.w.s.m.m.a.RequestMappingHandlerMapping : Mapped "{[/]}" onto public java.lang.String
com.tmall.ews.IndexController.index()
2021-07-19 16:38:27.756  INFO 9 --- [           main]
s.w.s.m.m.a.RequestMappingHandlerMapping : Mapped "{[/error],produces=[text/html]}" ont
o public org.springframework.web.servlet.ModelAndView
org.springframework.boot.autoconfigure.web.servlet.error.BasicErrorController.errorHtml(j
ax.servlet.http.HttpServletRequest,javax.servlet.http.HttpServletResponse)
2021-07-19 16:38:27.756  INFO 9 --- [           main]
s.w.s.m.m.a.RequestMappingHandlerMapping : Mapped "{[/error]}" onto public org.springfr
amework.http.ResponseEntity&lt;java.util.Map&lt;java.lang.String,
java.lang.Object&gt;&gt;
org.springframework.boot.autoconfigure.web.servlet.error.BasicErrorController.error(javax
ervlet.http.HttpServletRequest)
2021-07-19 16:38:27.761  INFO 9 --- [           main]
s.w.s.m.m.a.RequestMappingHandlerMapping : Mapped "{[/__mse__/readiness]}" onto public
org.springframework.http.ResponseEntity
com.navercorp.pinpoint.plugin.micro.service.interceptor.Mse_Readiness_Controller.healthCh
k()
2021-07-19 16:38:27.787  INFO 9 --- [           main]
o.s.w.s.handler.SimpleUrlHandlerMapping  : Mapped URL path [/webjars/**] onto handler o
f type [class org.springframework.web.servlet.resource.ResourceHttpRequestHandler]
2021-07-19 16:38:27.787  INFO 9 --- [           main]
o.s.w.s.handler.SimpleUrlHandlerMapping  : Mapped URL path [/**] onto handler of type [
class org.springframework.web.servlet.resource.ResourceHttpRequestHandler]
2021-07-19 16:38:28.291  INFO 9 --- [           main] o.s.j.e.a.AnnotationMBeanExporter
: Registering beans for JMX exposure on startup
2021-07-19 16:38:28.301  INFO 9 --- [           main]
ConfigServletWebServerApplicationContext : Refreshing
org.springframework.boot.web.servlet.context.AnnotationConfigServletWebServerApplicationC
text@5bb51241: startup date [Mon Jul 19 16:38:28 CST 2021]; parent:
org.springframework.boot.web.servlet.context.AnnotationConfigServletWebServerApplicationC
text@3d7cc3cb
2021-07-19 16:38:28.358  INFO 9 --- [           main]
o.s.b.f.s.DefaultListableBeanFactory     : Overriding bean definition for bean
'handlerExceptionResolver' with a different definition: replacing [Root bean: class [nu
ll]; scope=; abstract=false; lazyInit=false; autowireMode=3; dependencyCheck=0; autowir
eCandidate=true; primary=false;
factoryBeanName=org.springframework.web.servlet.config.annotation.DelegatingWebMvcConfigu
tion; factoryMethodName=handlerExceptionResolver; initMethodName=null;
destroyMethodName=(inferred); defined in class path resource
[org/springframework/web/servlet/config/annotation/DelegatingWebMvcConfiguration.class]]
with [Root bean: class [null]; scope=; abstract=false; lazyInit=false; autowireMode=3;
dependencyCheck=0; autowireCandidate=true; primary=false;
factoryBeanName=org.springframework.boot.actuate.autoconfigure.web.servlet.WebMvcEndpoint
ildContextConfiguration; factoryMethodName=compositeHandlerExceptionResolver; initMetho
dName=null; destroyMethodName=(inferred); defined in class path resource
[org/springframework/boot/actuate/autoconfigure/web/servlet/WebMvcEndpointChildContextCon
guration.class]]
2021-07-19 16:38:28.401  INFO 9 --- [           main]
o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat initialized with port(s): 8081 (http)
```

```
2021-07-19 16:38:28.403  INFO 9 --- [           main]
o.apache.catalina.core.StandardService   : Starting service [Tomcat]
2021-07-19 16:38:28.403  INFO 9 --- [           main]
org.apache.catalina.core.StandardEngine  : Starting Servlet Engine: Apache
Tomcat/8.5.31
2021-07-19 16:38:28.429  INFO 9 --- [ost-startStop-1] o.a.c.c.C.[Tomcat-1].[localhost].
[/]     : Initializing Spring embedded WebApplicationContext
2021-07-19 16:38:28.429  INFO 9 --- [ost-startStop-1] o.s.web.context.ContextLoader
: Root WebApplicationContext: initialization completed in 128 ms
2021-07-19 16:38:28.451  INFO 9 --- [ost-startStop-1]
o.s.b.w.servlet.ServletRegistrationBean  : Servlet dispatcherServlet mapped to [/]
2021-07-19 16:38:28.479  INFO 9 --- [           main]
o.s.b.a.e.web.EndpointLinksResolver      : Exposing 2 endpoint(s) beneath base path '/a
ctuator'
2021-07-19 16:38:28.492  INFO 9 --- [           main]
s.b.a.e.w.s.WebMvcEndpointHandlerMapping : Mapped "{[/actuator/health],methods=[GET],pr
oduces=[application/vnd.spring-boot.actuator.v2+json || application/json]}" onto public
java.lang.Object
org.springframework.boot.actuate.endpoint.web.servlet.AbstractWebMvcEndpointHandlerMappin
OperationHandler.handle(javax.servlet.http.HttpServletRequest,java.util.Map&lt;java.lang.
ring, java.lang.String&gt;)
2021-07-19 16:38:28.492  INFO 9 --- [           main]
s.b.a.e.w.s.WebMvcEndpointHandlerMapping : Mapped "{[/actuator/info],methods=[GET],prod
uces=[application/vnd.spring-boot.actuator.v2+json || application/json]}" onto public j
ava.lang.Object
org.springframework.boot.actuate.endpoint.web.servlet.AbstractWebMvcEndpointHandlerMappin
OperationHandler.handle(javax.servlet.http.HttpServletRequest,java.util.Map&lt;java.lang.
ring, java.lang.String&gt;)
2021-07-19 16:38:28.493  INFO 9 --- [           main]
s.b.a.e.w.s.WebMvcEndpointHandlerMapping : Mapped "{[/actuator],methods=[GET],produces=
[application/vnd.spring-boot.actuator.v2+json || application/json]}" onto protected jav
a.util.Map&lt;java.lang.String, java.util.Map&lt;java.lang.String,
org.springframework.boot.actuate.endpoint.web.Link&gt;&gt;
org.springframework.boot.actuate.endpoint.web.servlet.WebMvcEndpointHandlerMapping.links(
vax.servlet.http.HttpServletRequest,javax.servlet.http.HttpServletResponse)
2021-07-19 16:38:28.524  INFO 9 --- [           main]
s.w.s.m.m.a.RequestMappingHandlerMapping : Mapped "{[/error]}" onto public java.util.Ma
p&lt;java.lang.String, java.lang.Object&gt;
org.springframework.boot.actuate.autoconfigure.web.servlet.ManagementErrorEndpoint.invoke
rg.springframework.web.context.request.ServletWebRequest)
2021-07-19 16:38:28.529  INFO 9 --- [           main]
o.s.w.s.handler.SimpleUrlHandlerMapping  : Mapped URL path [/webjars/**] onto handler o
f type [class org.springframework.web.servlet.resource.ResourceHttpRequestHandler]
2021-07-19 16:38:28.529  INFO 9 --- [           main]
o.s.w.s.handler.SimpleUrlHandlerMapping  : Mapped URL path [/**] onto handler of type [
class org.springframework.web.servlet.resource.ResourceHttpRequestHandler]
2021-07-19 16:38:28.547  INFO 9 --- [           main]
s.w.s.m.m.a.RequestMappingHandlerAdapter : Looking for @ControllerAdvice:
org.springframework.boot.web.servlet.context.AnnotationConfigServletWebServerApplicationC
text@5bb51241: startup date [Mon Jul 19 16:38:28 CST 2021]; parent:
org.springframework.boot.web.servlet.context.AnnotationConfigServletWebServerApplicationC
text@3d7cc3cb
2021-07-19 16:38:28.598  INFO 9 --- [           main]
o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port(s): 8081 (http) with
```

```
context path ''
2021-07-19 16:38:28.620  INFO 9 --- [           main]
o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port(s): 8080 (http) with
context path ''
2021-07-19 16:38:28.621  INFO 9 --- [           main] com.tmall.ews.Application
: Started Application in 4.461 seconds (JVM running for 9.703)
</Content>
        </ContainerLogList>
        <ContainerLogList>
              <PodName>jck-deployment-yacs-24209-25073-728587-5757b8df59-rzpw6</PodName>
              <ContainerName>jck-sidecar-deploytool</ContainerName>
              <Content>Prepare Code Start
Step1: Delete /acs/code/*
Step2: Get User Code From URL http://yacs-share-zjk.oss-cn-zhangjiakou-
internal.aliyuncs.com/packet/lsy/24209/test/1624005052308/kq22elmv.war?
Expires=1634459880&amp;OSSAccessKeyId=RYswd40XbunQAtq4&amp;Signature=RLqIzheg4s1DdT4eOwdx
WWKCY%3D
http://yacs-share-zjk.oss-cn-zhangjiakou-
internal.aliyuncs.com/packet/lsy/24209/test/1624005052308/kq22elmv.war?
Expires=1634459880&amp;OSSAccessKeyId=RYswd40XbunQAtq4&amp;Signature=RLqIzheg4s1DdT4eOwdx
WWKCY%3D
Server:  172.25.0.10
Address: 172.25.0.10:53

*** Can't find yacs-share-zjk.oss-cn-zhangjiakou-internal.aliyuncs.com: No answer

*** Can't find yacs-share-zjk.oss-cn-zhangjiakou-internal.aliyuncs.com: No answer

need unzip code
true
Connecting to yacs-share-zjk.oss-cn-zhangjiakou-internal.aliyuncs.com
(100.118.90.245:80)
saving to 'code.zip'
code.zip              36% |***********                     | 6144k  0:00:01 ETA
code.zip             100% |********************************| 16.6M  0:00:00 ETA
'code.zip' saved
Archive:  code.zip
   creating: META-INF/
  inflating: META-INF/MANIFEST.MF
   creating: org/
   creating: org/springframework/
   creating: org/springframework/boot/
   creating: org/springframework/boot/loader/
   creating: org/springframework/boot/loader/data/
  inflating:
org/springframework/boot/loader/data/RandomAccessDataFile$DataInputStream.class
   creating: org/springframework/boot/loader/jar/
  inflating: org/springframework/boot/loader/jar/JarURLConnection.class
  inflating: org/springframework/boot/loader/jar/JarFile$JarFileType.class
  inflating: org/springframework/boot/loader/jar/JarFile.class
  inflating: org/springframework/boot/loader/jar/JarEntry.class
  inflating: org/springframework/boot/loader/jar/FileHeader.class
  inflating: org/springframework/boot/loader/jar/StringSequence.class
  inflating: org/springframework/boot/loader/jar/AsciiBytes.class
  inflating: org/springframework/boot/loader/jar/JarFileEntries$1.class
```

```
  inflating: org/springframework/boot/loader/jar/JarFileEntries$EntryIterator.class
  inflating: org/springframework/boot/loader/jar/JarFileEntries.class
  inflating: org/springframework/boot/loader/jar/CentralDirectoryVisitor.class
  inflating: org/springframework/boot/loader/jar/JarEntryFilter.class
  inflating: org/springframework/boot/loader/jar/CentralDirectoryFileHeader.class
  inflating: org/springframework/boot/loader/jar/CentralDirectoryEndRecord.class
  inflating: org/springframework/boot/loader/jar/CentralDirectoryParser.class
  inflating: org/springframework/boot/loader/jar/ZipInflaterInputStream.class
  inflating: org/springframework/boot/loader/jar/Handler.class
   creating: org/springframework/boot/loader/archive/
  inflating: org/springframework/boot/loader/archive/Archive$Entry.class
  inflating: org/springframework/boot/loader/archive/Archive$EntryFilter.class
  inflating: org/springframework/boot/loader/archive/Archive.class
  inflating:
org/springframework/boot/loader/archive/ExplodedArchive$FileEntryIterator$EntryComparator
lass
  inflating:
org/springframework/boot/loader/archive/ExplodedArchive$FileEntryIterator.class
  inflating: org/springframework/boot/loader/archive/ExplodedArchive.class
  inflating:
org/springframework/boot/loader/PropertiesLauncher$ArchiveEntryFilter.class
  inflating: org/springframework/boot/loader/PropertiesLauncher.class
  inflating: org/springframework/boot/loader/data/RandomAccessDataFile$1.class
  inflating: org/springframework/boot/loader/jar/JarFile$1.class
  inflating: org/springframework/boot/loader/jar/JarFile$2.class
  inflating: org/springframework/boot/loader/data/RandomAccessDataFile$FileAccess.class
  inflating: org/springframework/boot/loader/data/RandomAccessDataFile.class
  inflating: org/springframework/boot/loader/archive/ExplodedArchive$FileEntry.class
  inflating: org/springframework/boot/loader/archive/JarFileArchive$JarFileEntry.class
  inflating: org/springframework/boot/loader/archive/JarFileArchive.class
  inflating: org/springframework/boot/loader/MainMethodRunner.class
  inflating: org/springframework/boot/loader/ExecutableArchiveLauncher.class
   creating: org/springframework/boot/loader/util/
  inflating: org/springframework/boot/loader/util/SystemPropertyUtils.class
  inflating: org/springframework/boot/loader/data/RandomAccessData.class
  inflating: org/springframework/boot/loader/archive/ExplodedArchive$1.class
  inflating: org/springframework/boot/loader/archive/JarFileArchive$EntryIterator.class
  inflating:
org/springframework/boot/loader/PropertiesLauncher$PrefixMatchingArchiveFilter.class
  inflating: org/springframework/boot/loader/PropertiesLauncher$1.class
  inflating: org/springframework/boot/loader/Launcher.class
  inflating: org/springframework/boot/loader/WarLauncher.class
  inflating: org/springframework/boot/loader/JarLauncher.class
  inflating:
org/springframework/boot/loader/LaunchedURLClassLoader$UseFastConnectionExceptionsEnumera
on.class
  inflating: org/springframework/boot/loader/LaunchedURLClassLoader.class
  inflating: org/springframework/boot/loader/jar/Bytes.class
  inflating: org/springframework/boot/loader/jar/JarURLConnection$1.class
  inflating: org/springframework/boot/loader/jar/JarURLConnection$JarEntryName.class
   creating: WEB-INF/
   creating: WEB-INF/classes/
   creating: WEB-INF/classes/com/
   creating: WEB-INF/classes/com/tmall/
```

```
   creating: WEB-INF/classes/com/tmall/ews/
   creating: WEB-INF/lib/
  inflating: WEB-INF/classes/com/tmall/ews/IndexController.class
  inflating: WEB-INF/lib/log4j-to-slf4j-2.10.0.jar
  inflating: WEB-INF/lib/spring-jcl-5.0.6.RELEASE.jar
  inflating: WEB-INF/lib/jackson-annotations-2.9.0.jar
  inflating: WEB-INF/lib/logback-classic-1.2.3.jar
  inflating: WEB-INF/lib/LatencyUtils-2.0.3.jar
  inflating: WEB-INF/lib/micrometer-core-1.0.4.jar
  inflating: WEB-INF/lib/tomcat-embed-core-8.5.31.jar
  inflating: WEB-INF/classes/application.properties
  inflating: WEB-INF/lib/log4j-api-2.10.0.jar
  inflating: WEB-INF/lib/HdrHistogram-2.1.10.jar
  inflating: WEB-INF/lib/classmate-1.3.4.jar
  inflating: WEB-INF/lib/validation-api-2.0.1.Final.jar
  inflating: WEB-INF/lib/logback-core-1.2.3.jar
  inflating: WEB-INF/lib/spring-boot-starter-2.0.2.RELEASE.jar
  inflating: WEB-INF/lib/spring-boot-2.0.2.RELEASE.jar
  inflating: WEB-INF/classes/com/tmall/ews/Application.class
  inflating: WEB-INF/lib/spring-boot-starter-web-2.0.2.RELEASE.jar
  inflating: WEB-INF/lib/slf4j-api-1.7.25.jar
  inflating: WEB-INF/lib/jackson-datatype-jsr310-2.9.5.jar
  inflating: WEB-INF/lib/spring-boot-actuator-2.0.2.RELEASE.jar
  inflating: WEB-INF/lib/spring-expression-5.0.6.RELEASE.jar
  inflating: WEB-INF/lib/jboss-logging-3.3.2.Final.jar
  inflating: WEB-INF/lib/spring-webmvc-5.0.6.RELEASE.jar
  inflating: WEB-INF/lib/tomcat-embed-el-8.5.31.jar
  inflating: WEB-INF/lib/snakeyaml-1.19.jar
  inflating: WEB-INF/lib/jul-to-slf4j-1.7.25.jar
  inflating: WEB-INF/lib/spring-web-5.0.6.RELEASE.jar
  inflating: WEB-INF/lib/hibernate-validator-6.0.9.Final.jar
  inflating: WEB-INF/lib/spring-boot-starter-actuator-2.0.2.RELEASE.jar
  inflating: WEB-INF/lib/spring-aop-5.0.6.RELEASE.jar
  inflating: WEB-INF/lib/spring-boot-starter-tomcat-2.0.2.RELEASE.jar
  inflating: WEB-INF/lib/jackson-core-2.9.5.jar
  inflating: WEB-INF/lib/jackson-datatype-jdk8-2.9.5.jar
  inflating: WEB-INF/lib/javax.annotation-api-1.3.2.jar
  inflating: WEB-INF/lib/spring-boot-starter-logging-2.0.2.RELEASE.jar
  inflating: WEB-INF/lib/spring-boot-starter-json-2.0.2.RELEASE.jar
  inflating: WEB-INF/lib/spring-boot-actuator-autoconfigure-2.0.2.RELEASE.jar
  inflating: WEB-INF/lib/spring-boot-autoconfigure-2.0.2.RELEASE.jar
   creating: META-INF/maven/
   creating: META-INF/maven/com.tmall.ews/
   creating: META-INF/maven/com.tmall.ews/springboot2-demo/
  inflating: META-INF/maven/com.tmall.ews/springboot2-demo/pom.xml
  inflating: META-INF/maven/com.tmall.ews/springboot2-demo/pom.properties
  inflating: WEB-INF/lib/jackson-module-parameter-names-2.9.5.jar
  inflating: WEB-INF/lib/spring-beans-5.0.6.RELEASE.jar
  inflating: WEB-INF/lib/spring-context-5.0.6.RELEASE.jar
  inflating: WEB-INF/lib/spring-core-5.0.6.RELEASE.jar
  inflating: WEB-INF/lib/tomcat-embed-websocket-8.5.31.jar
  inflating: WEB-INF/lib/jackson-databind-2.9.5.jar
</Content>
        </ContainerLogList>
```

```
        <ContainerLogList>
              <PodName>jck-deployment-yacs-24209-25073-728587-5757b8df59-rzpw6</PodName>
              <ContainerName>arms-pilot-initcontainer</ContainerName>
              <Content>[init] Start arms k8s initer.
[REGIONID] current from env ARMS_PILOT_INIT_REGIONID: cn-zhangjiakou
I0719 08:38:12.783269       1 arms-k8s-init.go:107] Config:
{"PHPEnabled":false,"JavaEnabled":true,"LicenseKey":"","AppName":"","AppId":"ekkf778j1t@e
d65c86fc2c61","RegionId":"cn-zhangjiakou","UserId":"1479861961942385","Vpc":true}
Pilot Version: HEAD_6738422a_20210426110002

[init] RegionId:  cn-zhangjiakou
mkdir success!
rm success!
[AGENT DOWNLOAD URL] ENV ARMS_AGENT_DOWNLOAD_URL:
I0719 08:38:12.783698       1 arms-k8s-init.go:568] Obtain Grey Agent URL from:
%shttp://arms-dc-zb-internal.aliyuncs.com/agent/url?
pid=ekkf778j1t@e8bd65c86fc2c61&amp;region=cn-
zhangjiakou&amp;userId=1479861961942385&amp;appId=ekkf778j1t@e8bd65c86fc2c61&amp;requestI
XVlBzgbaiC
E0719 08:38:12.837189       1 arms-k8s-init.go:586] javaGreyUrl unmarshal json: cannot
unmarshal object into Go struct field Response.Data of type string
Grey URL:
No Grey Agent
[getAgentDownloadUrlByRegion]: http://arms-apm-zhangjiakou.oss-cn-zhangjiakou-internal.
aliyuncs.com/
[AgentDownloadUrl] final url: http://arms-apm-zhangjiakou.oss-cn-zhangjiakou-internal.a
liyuncs.com/, isCopyAgentConfig: false, isFullAgentUrl: false
[getAgentDownloadUrlByRegion]: http://arms-apm-zhangjiakou.oss-cn-zhangjiakou-internal.
aliyuncs.com/
[init] Try download arms agent. agentUrl: http://arms-apm-zhangjiakou.oss-cn-zhangjiako
u-internal.aliyuncs.com/ArmsAgent.zip, http://arms-apm-zhangjiakou.oss-cn-zhangjiakou-i
nternal.aliyuncs.com/Arthas.zip, nowTime: 2021-07-19 08:38:12.837238928 +0000 UTC m=+0.
063203723
[AgentDownloadTimeoutSec] ENV ARMS_AGENT_DOWNLOAD_TIMEOUT_SEC:
[ARMS_PILOT_INIT_CACHE_MODE_MANUEL] ENV ARMS_PILOT_INIT_CACHE_MODE:
[Download JAVA Agent] success. cost time: 1
unzip /home/admin/.opt/, costTime: 3
unzip /home/admin/.opt/, costTime: 1
[originalConfigSwitch] skip copy region agent config. use the agent original config.
[chmod] success. dirPath: /home/admin/.opt/ArmsAgent
[chmod] success. dirPath: /home/admin/.opt/Arthas
[Unzip JAVA Agent] success. cost time: 5
DONE
</Content>
        </ContainerLogList>
  </Result>
</DescribePodContainerLogList>
```

`JSON` **格式**

```
{
    "RequestId": "F2053086-E9E0-1164-B05A-9CF5C79BDAAF",
    "Code": 0,
    "Success": true,
```

```
    "Result": {
        "ContainerLogList": [
            {
                "PodName": "jck-deployment-yacs-24209-25073-728587-5757b8df59-rzpw6",
                "ContainerName": "jck-app-container-24209-25073",
                "Content": "-server -Xms2g -Xmx2g -Xmn1g -XX:MetaspaceSize=256m -XX:MaxM
etaspaceSize=256m -Duser.timezone=Asia/Shanghai\n--management.info.build.mode=full --sp
ring.profiles.active=production --logging.path=/acs/log --
logging.file=/acs/log/application.log\n8080\n8081\nwar inf dir exist\nPicked up JAVA_TO
OL_OPTIONS:  -Doneagent.plugin.arms-agent.enabled=true -
javaagent:/home/admin/.opt/ArmsAgent/arms-bootstrap-1.7.0-SNAPSHOT.jar -
Darms.licenseKey=ekkf778j1t@d643cdcd427b3bf -Darms.appId=ekkf778j1t@e8bd65c86fc2c61 -Da
rms.agent.env=ACSK8S  -Darms.agent.podinfo.path=/etc/podinfo\nSLF4J: Failed to load cla
ss \"org.slf4j.impl.StaticLoggerBinder\".\nSLF4J: Defaulting to no-operation (NOP) logg
er implementation\nSLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for fu
rther details.\n2021-07-19 16:38:23 [WARN ]
(com.navercorp.pinpoint.bootstrap.PinpointBootStrap) arms agent start cost: 4607ms\n\n
.   ____          _            __ _ _\n /\\\\ / ___'_ __ _ _(_)_ __  __ _ \\ \\ \\ \\\\n(
( )\\___ | '_ | '_| | '_ \\/ _` | \\ \\ \\ \\\\n \\\\/  ___)| |_)| | | | | || (_| |  ) )
) )\n  '  |____| .__|_| |_|_| |_\\__, | / / / /\n
=========|_|==============|___/=/_/_/_/\n :: Spring Boot ::
(v2.0.2.RELEASE)\n\n2021-07-19 16:38:24.932  INFO 9 --- [           main]
com.tmall.ews.Application                : Starting Application on jck-deployment-yacs-2
4209-25073-728587-5757b8df59-rzpw6 with PID 9 (/acs/code/WEB-INF/classes started by roo
t in /acs/code)\n2021-07-19 16:38:24.933  INFO 9 --- [           main] com.tmall.ews.App
lication                : The following profiles are active: production\n2021-07-19
16:38:25.005  INFO 9 --- [           main] ConfigServletWebServerApplicationContext :
Refreshing
org.springframework.boot.web.servlet.context.AnnotationConfigServletWebServerApplicationC
text@3d7cc3cb: startup date [Mon Jul 19 16:38:25 CST 2021]; root of context hierarchy\n
2021-07-19 16:38:26.706  INFO 9 --- [           main]
o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat initialized with port(s): 8080 (http)
\n2021-07-19 16:38:26.742  INFO 9 --- [           main]
o.apache.catalina.core.StandardService   : Starting service [Tomcat]\n2021-07-19 16:38:
26.742  INFO 9 --- [           main] org.apache.catalina.core.StandardEngine  :
Starting Servlet Engine: Apache Tomcat/8.5.31\n2021-07-19 16:38:26.754  INFO 9 --- [ost
-startStop-1] o.a.catalina.core.AprLifecycleListener   : The APR based Apache Tomcat Na
tive library which allows optimal performance in production environments was not found
on the java.library.path:
[/usr/java/packages/lib/amd64:/usr/lib64:/lib64:/lib:/usr/lib]\n2021-07-19 16:38:26.869
INFO 9 --- [ost-startStop-1] o.a.c.c.C.[Tomcat].[localhost].[/]       : Initializing Sp
ring embedded WebApplicationContext\n2021-07-19 16:38:26.869  INFO 9 --- [ost-startStop
-1] o.s.web.context.ContextLoader            : Root WebApplicationContext:
initialization completed in 1864 ms\n2021-07-19 16:38:27.194  INFO 9 --- [ost-
startStop-1] o.s.b.w.servlet.ServletRegistrationBean  : Servlet dispatcherServlet mappe
d to [/]\n2021-07-19 16:38:27.272  INFO 9 --- [ost-startStop-1]
o.s.b.w.servlet.FilterRegistrationBean   : Mapping filter: 'characterEncodingFilter' to
: [/*]\n2021-07-19 16:38:27.273  INFO 9 --- [ost-startStop-1]
o.s.b.w.servlet.FilterRegistrationBean   : Mapping filter: 'hiddenHttpMethodFilter' to:
[/*]\n2021-07-19 16:38:27.273  INFO 9 --- [ost-startStop-1]
o.s.b.w.servlet.FilterRegistrationBean   : Mapping filter: 'httpPutFormContentFilter' t
o: [/*]\n2021-07-19 16:38:27.273  INFO 9 --- [ost-startStop-1]
o.s.b.w.servlet.FilterRegistrationBean   : Mapping filter: 'requestContextFilter' to: [
/*]\n2021-07-19 16:38:27.273  INFO 9 --- [ost-startStop-1]
```

```
o.s.b.w.servlet.FilterRegistrationBean   : Mapping filter: 'httpTraceFilter' to: [/*]\n
2021-07-19 16:38:27.274  INFO 9 --- [ost-startStop-1]
o.s.b.w.servlet.FilterRegistrationBean   : Mapping filter: 'webMvcMetricsFilter' to: [/
*]\n2021-07-19 16:38:27.423  INFO 9 --- [           main]
o.s.w.s.handler.SimpleUrlHandlerMapping  : Mapped URL path [/**/favicon.ico] onto handl
er of type [class
org.springframework.web.servlet.resource.ResourceHttpRequestHandler]\n2021-07-19 16:38:
27.653  INFO 9 --- [           main] s.w.s.m.m.a.RequestMappingHandlerAdapter : Looking
for @ControllerAdvice:
org.springframework.boot.web.servlet.context.AnnotationConfigServletWebServerApplicationC
text@3d7cc3cb: startup date [Mon Jul 19 16:38:25 CST 2021]; root of context hierarchy\n
2021-07-19 16:38:27.750  INFO 9 --- [           main]
s.w.s.m.m.a.RequestMappingHandlerMapping : Mapped \"{[/__mse__/readiness]}\" onto publi
c org.springframework.http.ResponseEntity
com.navercorp.pinpoint.plugin.micro.service.interceptor.Mse_Readiness_Controller.healthCh
k()\n2021-07-19 16:38:27.752  INFO 9 --- [           main]
s.w.s.m.m.a.RequestMappingHandlerMapping : Mapped \"{[/]}\" onto public
java.lang.String com.tmall.ews.IndexController.index()\n2021-07-19 16:38:27.756  INFO 9
--- [           main] s.w.s.m.m.a.RequestMappingHandlerMapping : Mapped \"{[/error],prod
uces=[text/html]}\" onto public org.springframework.web.servlet.ModelAndView org.spring
framework.boot.autoconfigure.web.servlet.error.BasicErrorController.errorHtml(javax.servl
.http.HttpServletRequest,javax.servlet.http.HttpServletResponse)\n2021-07-19 16:38:27.7
56  INFO 9 --- [           main] s.w.s.m.m.a.RequestMappingHandlerMapping : Mapped \"
{[/error]}\" onto public
org.springframework.http.ResponseEntity<java.util.Map<java.lang.String,
java.lang.Object>>
org.springframework.boot.autoconfigure.web.servlet.error.BasicErrorController.error(javax
ervlet.http.HttpServletRequest)\n2021-07-19 16:38:27.761  INFO 9 --- [           main]
s.w.s.m.m.a.RequestMappingHandlerMapping : Mapped \"{[/__mse__/readiness]}\" onto publi
c org.springframework.http.ResponseEntity
com.navercorp.pinpoint.plugin.micro.service.interceptor.Mse_Readiness_Controller.healthCh
k()\n2021-07-19 16:38:27.787  INFO 9 --- [           main]
o.s.w.s.handler.SimpleUrlHandlerMapping  : Mapped URL path [/webjars/**] onto handler o
f type [class
org.springframework.web.servlet.resource.ResourceHttpRequestHandler]\n2021-07-19 16:38:
27.787  INFO 9 --- [           main] o.s.w.s.handler.SimpleUrlHandlerMapping  : Mapped
URL path [/**] onto handler of type [class
org.springframework.web.servlet.resource.ResourceHttpRequestHandler]\n2021-07-19 16:38:
28.291  INFO 9 --- [           main] o.s.j.e.a.AnnotationMBeanExporter        : Register
ing beans for JMX exposure on startup\n2021-07-19 16:38:28.301  INFO 9 --- [
main] ConfigServletWebServerApplicationContext : Refreshing
org.springframework.boot.web.servlet.context.AnnotationConfigServletWebServerApplicationC
text@5bb51241: startup date [Mon Jul 19 16:38:28 CST 2021]; parent:
org.springframework.boot.web.servlet.context.AnnotationConfigServletWebServerApplicationC
text@3d7cc3cb\n2021-07-19 16:38:28.358  INFO 9 --- [           main]
o.s.b.f.s.DefaultListableBeanFactory     : Overriding bean definition for bean
'handlerExceptionResolver' with a different definition: replacing [Root bean: class [nu
ll]; scope=; abstract=false; lazyInit=false; autowireMode=3; dependencyCheck=0; autowir
eCandidate=true; primary=false;
factoryBeanName=org.springframework.web.servlet.config.annotation.DelegatingWebMvcConfigu
tion; factoryMethodName=handlerExceptionResolver; initMethodName=null;
destroyMethodName=(inferred); defined in class path resource
[org/springframework/web/servlet/config/annotation/DelegatingWebMvcConfiguration.class]]
with [Root bean: class [null]; scope=; abstract=false; lazyInit=false; autowireMode=3;
```

```
dependencyCheck=0; autowireCandidate=true; primary=false;
factoryBeanName=org.springframework.boot.actuate.autoconfigure.web.servlet.WebMvcEndpoint
ildContextConfiguration; factoryMethodName=compositeHandlerExceptionResolver; initMetho
dName=null; destroyMethodName=(inferred); defined in class path resource
[org/springframework/boot/actuate/autoconfigure/web/servlet/WebMvcEndpointChildContextCon
guration.class]]\n2021-07-19 16:38:28.401  INFO 9 --- [           main]
o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat initialized with port(s): 8081 (http)
\n2021-07-19 16:38:28.403  INFO 9 --- [           main]
o.apache.catalina.core.StandardService   : Starting service [Tomcat]\n2021-07-19 16:38:
28.403  INFO 9 --- [           main] org.apache.catalina.core.StandardEngine  :
Starting Servlet Engine: Apache Tomcat/8.5.31\n2021-07-19 16:38:28.429  INFO 9 --- [ost
-startStop-1] o.a.c.c.C.[Tomcat-1].[localhost].[/]     : Initializing Spring embedded W
ebApplicationContext\n2021-07-19 16:38:28.429  INFO 9 --- [ost-startStop-1]
o.s.web.context.ContextLoader            : Root WebApplicationContext: initialization
completed in 128 ms\n2021-07-19 16:38:28.451  INFO 9 --- [ost-startStop-1]
o.s.b.w.servlet.ServletRegistrationBean  : Servlet dispatcherServlet mapped to [/]\n202
1-07-19 16:38:28.479  INFO 9 --- [           main] o.s.b.a.e.web.EndpointLinksResolver
: Exposing 2 endpoint(s) beneath base path '/actuator'\n2021-07-19 16:38:28.492  INFO 9
--- [           main] s.b.a.e.w.s.WebMvcEndpointHandlerMapping : Mapped \"{[/actuator/he
alth],methods=[GET],produces=[application/vnd.spring-boot.actuator.v2+json ||
application/json]}\" onto public java.lang.Object
org.springframework.boot.actuate.endpoint.web.servlet.AbstractWebMvcEndpointHandlerMappin
OperationHandler.handle(javax.servlet.http.HttpServletRequest,java.util.Map<java.lang.Str
g, java.lang.String>)\n2021-07-19 16:38:28.492  INFO 9 --- [           main]
s.b.a.e.w.s.WebMvcEndpointHandlerMapping : Mapped \"{[/actuator/info],methods=[GET],pro
duces=[application/vnd.spring-boot.actuator.v2+json || application/json]}\" onto public
java.lang.Object
org.springframework.boot.actuate.endpoint.web.servlet.AbstractWebMvcEndpointHandlerMappin
OperationHandler.handle(javax.servlet.http.HttpServletRequest,java.util.Map<java.lang.Str
g, java.lang.String>)\n2021-07-19 16:38:28.493  INFO 9 --- [           main]
s.b.a.e.w.s.WebMvcEndpointHandlerMapping : Mapped \"{[/actuator],methods=
[GET],produces=[application/vnd.spring-boot.actuator.v2+json || application/json]}\" on
to protected java.util.Map<java.lang.String, java.util.Map<java.lang.String, org.spring
framework.boot.actuate.endpoint.web.Link>>
org.springframework.boot.actuate.endpoint.web.servlet.WebMvcEndpointHandlerMapping.links(
vax.servlet.http.HttpServletRequest,javax.servlet.http.HttpServletResponse)\n2021-07-19
16:38:28.524  INFO 9 --- [           main] s.w.s.m.m.a.RequestMappingHandlerMapping :
Mapped \"{[/error]}\" onto public java.util.Map<java.lang.String, java.lang.Object> org
.springframework.boot.actuate.autoconfigure.web.servlet.ManagementErrorEndpoint.invoke(or
springframework.web.context.request.ServletWebRequest)\n2021-07-19 16:38:28.529  INFO 9
--- [           main] o.s.w.s.handler.SimpleUrlHandlerMapping  : Mapped URL path
[/webjars/**] onto handler of type [class
org.springframework.web.servlet.resource.ResourceHttpRequestHandler]\n2021-07-19 16:38:
28.529  INFO 9 --- [           main] o.s.w.s.handler.SimpleUrlHandlerMapping  : Mapped
URL path [/**] onto handler of type [class
org.springframework.web.servlet.resource.ResourceHttpRequestHandler]\n2021-07-19 16:38:
28.547  INFO 9 --- [           main] s.w.s.m.m.a.RequestMappingHandlerAdapter : Looking
for @ControllerAdvice:
org.springframework.boot.web.servlet.context.AnnotationConfigServletWebServerApplicationC
text@5bb51241: startup date [Mon Jul 19 16:38:28 CST 2021]; parent:
org.springframework.boot.web.servlet.context.AnnotationConfigServletWebServerApplicationC
text@3d7cc3cb\n2021-07-19 16:38:28.598  INFO 9 --- [           main]
o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port(s): 8081 (http) with
context path ''\n2021-07-19 16:38:28.620  INFO 9 --- [           main]
```

```
o.s.b.w.embedded.tomcat.TomcatWebServer  : Tomcat started on port(s): 8080 (http) with
context path ''\n2021-07-19 16:38:28.621  INFO 9 --- [           main]
com.tmall.ews.Application                : Started Application in 4.461 seconds (JVM run
ning for 9.703)\n"
            },
            {
                "PodName": "jck-deployment-yacs-24209-25073-728587-5757b8df59-rzpw6",
                "ContainerName": "jck-sidecar-deploytool",
                "Content": "Prepare Code Start\nStep1: Delete /acs/code/*\nStep2: Get Us
er Code From URL http://yacs-share-zjk.oss-cn-zhangjiakou-
internal.aliyuncs.com/packet/lsy/24209/test/1624005052308/kq22elmv.war?
Expires=1634459880&OSSAccessKeyId=RYswd40XbunQAtq4&Signature=RLqIzheg4s1DdT4eOwdxviWWKCY%
\nhttp://yacs-share-zjk.oss-cn-zhangjiakou-
internal.aliyuncs.com/packet/lsy/24209/test/1624005052308/kq22elmv.war?
Expires=1634459880&OSSAccessKeyId=RYswd40XbunQAtq4&Signature=RLqIzheg4s1DdT4eOwdxviWWKCY%
\nServer:\t\t172.25.0.10\nAddress:\t172.25.0.10:53\n\n*** Can't find yacs-share-zjk.oss
-cn-zhangjiakou-internal.aliyuncs.com: No answer\n\n*** Can't find yacs-share-zjk.oss-c
n-zhangjiakou-internal.aliyuncs.com: No answer\n\nneed unzip code\ntrue\nConnecting to
yacs-share-zjk.oss-cn-zhangjiakou-internal.aliyuncs.com (100.118.90.245:80)\nsaving to
'code.zip'\ncode.zip              36% |***********                     | 6144k  0:00:01
ETA\ncode.zip             100% |********************************| 16.6M  0:00:00 ETA\n'c
ode.zip' saved\nArchive:  code.zip\n   creating: META-INF/\n  inflating: META-
INF/MANIFEST.MF\n   creating: org/\n   creating: org/springframework/\n   creating: org
/springframework/boot/\n   creating: org/springframework/boot/loader/\n   creating: org
/springframework/boot/loader/data/\n  inflating:
org/springframework/boot/loader/data/RandomAccessDataFile$DataInputStream.class\n   cre
ating: org/springframework/boot/loader/jar/\n  inflating:
org/springframework/boot/loader/jar/JarURLConnection.class\n  inflating:
org/springframework/boot/loader/jar/JarFile$JarFileType.class\n  inflating:
org/springframework/boot/loader/jar/JarFile.class\n  inflating:
org/springframework/boot/loader/jar/JarEntry.class\n  inflating:
org/springframework/boot/loader/jar/FileHeader.class\n  inflating:
org/springframework/boot/loader/jar/StringSequence.class\n  inflating:
org/springframework/boot/loader/jar/AsciiBytes.class\n  inflating:
org/springframework/boot/loader/jar/JarFileEntries$1.class\n  inflating:
org/springframework/boot/loader/jar/JarFileEntries$EntryIterator.class\n  inflating: or
g/springframework/boot/loader/jar/JarFileEntries.class\n  inflating:
org/springframework/boot/loader/jar/CentralDirectoryVisitor.class\n  inflating: org/spr
ingframework/boot/loader/jar/JarEntryFilter.class\n  inflating:
org/springframework/boot/loader/jar/CentralDirectoryFileHeader.class\n  inflating: org/
springframework/boot/loader/jar/CentralDirectoryEndRecord.class\n  inflating:
org/springframework/boot/loader/jar/CentralDirectoryParser.class\n  inflating: org/spri
ngframework/boot/loader/jar/ZipInflaterInputStream.class\n  inflating:
org/springframework/boot/loader/jar/Handler.class\n   creating:
org/springframework/boot/loader/archive/\n  inflating:
org/springframework/boot/loader/archive/Archive$Entry.class\n  inflating:
org/springframework/boot/loader/archive/Archive$EntryFilter.class\n  inflating: org/spr
ingframework/boot/loader/archive/Archive.class\n  inflating:
org/springframework/boot/loader/archive/ExplodedArchive$FileEntryIterator$EntryComparator
lass\n  inflating:
org/springframework/boot/loader/archive/ExplodedArchive$FileEntryIterator.class\n  infl
ating: org/springframework/boot/loader/archive/ExplodedArchive.class\n  inflating: org/
springframework/boot/loader/PropertiesLauncher$ArchiveEntryFilter.class\n  inflating: o
rg/springframework/boot/loader/PropertiesLauncher.class\n  inflating:
org/springframework/boot/loader/data/RandomAccessDataFile$1.class\n  inflating: org/spr
```

```
org/springframework/boot/loader/data/RandomAccessDataFile$1.class\n  inflating: org/spr
ingframework/boot/loader/jar/JarFile$1.class\n  inflating:
org/springframework/boot/loader/jar/JarFile$2.class\n  inflating:
org/springframework/boot/loader/data/RandomAccessDataFile$FileAccess.class\n  inflating
: org/springframework/boot/loader/data/RandomAccessDataFile.class\n  inflating: org/spr
ingframework/boot/loader/archive/ExplodedArchive$FileEntry.class\n  inflating: org/spri
ngframework/boot/loader/archive/JarFileArchive$JarFileEntry.class\n  inflating: org/spr
ingframework/boot/loader/archive/JarFileArchive.class\n  inflating:
org/springframework/boot/loader/MainMethodRunner.class\n  inflating:
org/springframework/boot/loader/ExecutableArchiveLauncher.class\n   creating:
org/springframework/boot/loader/util/\n  inflating:
org/springframework/boot/loader/util/SystemPropertyUtils.class\n  inflating:
org/springframework/boot/loader/data/RandomAccessData.class\n  inflating:
org/springframework/boot/loader/archive/ExplodedArchive$1.class\n  inflating:
org/springframework/boot/loader/archive/JarFileArchive$EntryIterator.class\n  inflating
:
org/springframework/boot/loader/PropertiesLauncher$PrefixMatchingArchiveFilter.class\n
inflating: org/springframework/boot/loader/PropertiesLauncher$1.class\n  inflating: org
/springframework/boot/loader/Launcher.class\n  inflating:
org/springframework/boot/loader/WarLauncher.class\n  inflating:
org/springframework/boot/loader/JarLauncher.class\n  inflating:
org/springframework/boot/loader/LaunchedURLClassLoader$UseFastConnectionExceptionsEnumera
on.class\n  inflating: org/springframework/boot/loader/LaunchedURLClassLoader.class\n
inflating: org/springframework/boot/loader/jar/Bytes.class\n  inflating:
org/springframework/boot/loader/jar/JarURLConnection$1.class\n  inflating:
org/springframework/boot/loader/jar/JarURLConnection$JarEntryName.class\n   creating: W
EB-INF/\n   creating: WEB-INF/classes/\n   creating: WEB-INF/classes/com/\n   creating:
WEB-INF/classes/com/tmall/\n   creating: WEB-INF/classes/com/tmall/ews/\n   creating: W
EB-INF/lib/\n  inflating: WEB-INF/classes/com/tmall/ews/IndexController.class\n  inflat
ing: WEB-INF/lib/log4j-to-slf4j-2.10.0.jar\n  inflating: WEB-INF/lib/spring-jcl-5.0.6.R
ELEASE.jar\n  inflating: WEB-INF/lib/jackson-annotations-2.9.0.jar\n  inflating: WEB-IN
F/lib/logback-classic-1.2.3.jar\n  inflating: WEB-INF/lib/LatencyUtils-2.0.3.jar\n  inf
lating: WEB-INF/lib/micrometer-core-1.0.4.jar\n  inflating: WEB-INF/lib/tomcat-embed-co
re-8.5.31.jar\n  inflating: WEB-INF/classes/application.properties\n  inflating: WEB-IN
F/lib/log4j-api-2.10.0.jar\n  inflating: WEB-INF/lib/HdrHistogram-2.1.10.jar\n  inflati
ng: WEB-INF/lib/classmate-1.3.4.jar\n  inflating: WEB-INF/lib/validation-api-
2.0.1.Final.jar\n  inflating: WEB-INF/lib/logback-core-1.2.3.jar\n  inflating: WEB-INF/
lib/spring-boot-starter-2.0.2.RELEASE.jar\n  inflating: WEB-INF/lib/spring-boot-2.0.2.R
ELEASE.jar\n  inflating: WEB-INF/classes/com/tmall/ews/Application.class\n  inflating:
WEB-INF/lib/spring-boot-starter-web-2.0.2.RELEASE.jar\n  inflating: WEB-INF/lib/slf4j-a
pi-1.7.25.jar\n  inflating: WEB-INF/lib/jackson-datatype-jsr310-2.9.5.jar\n  inflating:
WEB-INF/lib/spring-boot-actuator-2.0.2.RELEASE.jar\n  inflating: WEB-INF/lib/spring-exp
ression-5.0.6.RELEASE.jar\n  inflating: WEB-INF/lib/jboss-logging-3.3.2.Final.jar\n  in
flating: WEB-INF/lib/spring-webmvc-5.0.6.RELEASE.jar\n  inflating: WEB-INF/lib/tomcat-e
mbed-el-8.5.31.jar\n  inflating: WEB-INF/lib/snakeyaml-1.19.jar\n  inflating: WEB-INF/l
ib/jul-to-slf4j-1.7.25.jar\n  inflating: WEB-INF/lib/spring-web-5.0.6.RELEASE.jar\n  in
flating: WEB-INF/lib/hibernate-validator-6.0.9.Final.jar\n  inflating: WEB-
INF/lib/spring-boot-starter-actuator-2.0.2.RELEASE.jar\n  inflating: WEB-
INF/lib/spring-aop-5.0.6.RELEASE.jar\n  inflating: WEB-INF/lib/spring-boot-starter-tomc
at-2.0.2.RELEASE.jar\n  inflating: WEB-INF/lib/jackson-core-2.9.5.jar\n  inflating: WEB
-INF/lib/jackson-datatype-jdk8-2.9.5.jar\n  inflating: WEB-INF/lib/javax.annotation-api
-1.3.2.jar\n  inflating: WEB-INF/lib/spring-boot-starter-logging-2.0.2.RELEASE.jar\n  i
nflating: WEB-INF/lib/spring-boot-starter-json-2.0.2.RELEASE.jar\n  inflating: WEB-INF/
lib/spring-boot-actuator-autoconfigure-2.0.2.RELEASE.jar\n  inflating: WEB-
INF/lib/spring-boot-autoconfigure-2.0.2.RELEASE.jar\n   creating: META-INF/maven/\n   c
```

```
reating: META-INF/maven/com.tmall.ews/\n   creating: META-
INF/maven/com.tmall.ews/springboot2-demo/\n  inflating: META-
INF/maven/com.tmall.ews/springboot2-demo/pom.xml\n  inflating: META-
INF/maven/com.tmall.ews/springboot2-demo/pom.properties\n  inflating: WEB-
INF/lib/jackson-module-parameter-names-2.9.5.jar\n  inflating: WEB-INF/lib/spring-beans
-5.0.6.RELEASE.jar\n  inflating: WEB-INF/lib/spring-context-5.0.6.RELEASE.jar\n
inflating: WEB-INF/lib/spring-core-5.0.6.RELEASE.jar\n  inflating: WEB-INF/lib/tomcat-e
mbed-websocket-8.5.31.jar\n  inflating: WEB-INF/lib/jackson-databind-2.9.5.jar\n"
            },
            {
                "PodName": "jck-deployment-yacs-24209-25073-728587-5757b8df59-rzpw6",
                "ContainerName": "arms-pilot-initcontainer",
                "Content": "[init] Start arms k8s initer.\n[REGIONID] current from env A
RMS_PILOT_INIT_REGIONID: cn-zhangjiakou\nI0719 08:38:12.783269       1 arms-k8s-
init.go:107] Config:
{\"PHPEnabled\":false,\"JavaEnabled\":true,\"LicenseKey\":\"\",\"AppName\":\"\",\"AppId\"
"ekkf778j1t@e8bd65c86fc2c61\",\"RegionId\":\"cn-
zhangjiakou\",\"UserId\":\"1479861961942385\",\"Vpc\":true}\nPilot Version: HEAD_673842
2a_20210426110002\n\n[init] RegionId:  cn-zhangjiakou\nmkdir success!\nrm
success!\n[AGENT DOWNLOAD URL] ENV ARMS_AGENT_DOWNLOAD_URL: \nI0719 08:38:12.783698
1 arms-k8s-init.go:568] Obtain Grey Agent URL from: %shttp://arms-dc-zb-
internal.aliyuncs.com/agent/url?pid=ekkf778j1t@e8bd65c86fc2c61&region=cn-
zhangjiakou&userId=1479861961942385&appId=ekkf778j1t@e8bd65c86fc2c61&requestId=XVlBzgbaiC
E0719 08:38:12.837189       1 arms-k8s-init.go:586] javaGreyUrl unmarshal json: cannot
unmarshal object into Go struct field Response.Data of type string\nGrey URL:  \nNo Gre
y Agent\n[getAgentDownloadUrlByRegion]: http://arms-apm-zhangjiakou.oss-cn-zhangjiakou-
internal.aliyuncs.com/\n[AgentDownloadUrl] final url: http://arms-apm-zhangjiakou.oss-c
n-zhangjiakou-internal.aliyuncs.com/, isCopyAgentConfig: false, isFullAgentUrl: false\n
[getAgentDownloadUrlByRegion]: http://arms-apm-zhangjiakou.oss-cn-zhangjiakou-internal.
aliyuncs.com/\n[init] Try download arms agent. agentUrl: http://arms-apm-zhangjiakou.os
s-cn-zhangjiakou-internal.aliyuncs.com/ArmsAgent.zip, http://arms-apm-zhangjiakou.oss-c
n-zhangjiakou-internal.aliyuncs.com/Arthas.zip, nowTime: 2021-07-19 08:38:12.837238928
+0000 UTC m=+0.063203723\n[AgentDownloadTimeoutSec] ENV
ARMS_AGENT_DOWNLOAD_TIMEOUT_SEC: \n[ARMS_PILOT_INIT_CACHE_MODE_MANUEL] ENV ARMS_PILOT_I
NIT_CACHE_MODE: \n[Download JAVA Agent] success. cost time: 1\nunzip /home/admin/.opt/,
costTime: 3\nunzip /home/admin/.opt/, costTime: 1\n[originalConfigSwitch] skip copy reg
ion agent config. use the agent original config.\n[chmod] success. dirPath: /home/admin
/.opt/ArmsAgent\n[chmod] success. dirPath: /home/admin/.opt/Arthas\n[Unzip JAVA Agent]
success. cost time: 5\nDONE\n"
            }
        ]
    }
}
```


错误码

HttpCode	错误码	错误信息	描述
500	system.error	A system error occurred.	系统异常

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

4.2. DescribeEventMonitorList

调用DescribeEventMonitorList查询事件监控。该接口查询从SLS事件中心查询，前提是开通了事件监控

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	DescribeEventMonitorList	系统规定参数。取值：DescribeEventMonitorList。
AppId	Long	是	1122	云应用id
EndTime	Long	是	1627484424000	查询截止时间
EnvId	Long	是	1122	云应用环境id
EventLevel	String	是	Normal	非必填，事件等级，Normal | Warning | Error
EventType	String	是	POD_FAILED	非必填。枚举值：POD_FAILED | READINESS_FAILED | LIVENESS_FAILED | EVICTED_BY_NODE | RESOURCE_INSUFFICIENT
PageNum	Integer	是	1	分页数
PageSize	Integer	是	20	分页大小
PodName	String	是	xxx	实例名称
StartTime	Long	是	1627484424000	查询开始时间

返回数据

名称	类型	示例值	描述
Code	Integer	0	状态码

Data	Array of EventInfo		事件列表
Count	Integer	1	事件发生次数
EventTime	String	2021-07-28T09:16Z	事件发生时间
HostName	String	cn-zhangjiakou.192.168.0.1	集群节点名称
Kind	String	xxx	类型 Pod | Node
Level	String	xxxxx	事件等级
Message	String	xxxx	事件详情
NameSpace	String	xxx	命名空间
PodName	String	xxx	pod实例名称
Reason	String	FailedScheduling	原因
ErrorMsg	String	xxx	错误信息
PageNumber	Integer	1	分页数
PageSize	Integer	20	分页大小
RequestId	String	xxx-xxx	请求id
TotalCount	Long	100	总数

示例

请求示例

```
http(s)://[Endpoint]/?Action=DescribeEventMonitorList
&AppId=1122
&EndTime=1627484424000
&EnvId=1122
&EventLevel=Normal
&EventType=POD_FAILED
&PageNum=1
&PageSize=20
&PodName=xxx
&StartTime=1627484424000
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<DescribeEventMonitorList>
  <TotalCount>12</TotalCount>
  <PageSize>20</PageSize>
  <RequestId>A1274507-33EE-13DD-A1EB-9B55F43E2F9E</RequestId>
  <PageNumber>1</PageNumber>
  <Data>
        <PodName>jck-deployment-yacs-24209-31780-748808-6d4cc4bc4-tbm4j</PodName>
        <Message>Started container jck-app-container-24209-31780</Message>
        <EventTime>2021-07-28T09:16Z</EventTime>
        <NameSpace>jck-namespace-24209</NameSpace>
        <Level>Normal</Level>
        <HostName>eci</HostName>
        <Reason>Started</Reason>
  </Data>
  <Data>
        <PodName>jck-deployment-yacs-24209-31780-748808-6d4cc4bc4-tbm4j</PodName>
        <Message>0/2 nodes are available: 1 Insufficient cpu, 1 node(s) had taint
{virtual-kubelet.io/provider: alibabacloud}, that the pod didn't tolerate.</Message>
        <EventTime>2021-07-28T09:15Z</EventTime>
        <NameSpace>jck-namespace-24209</NameSpace>
        <Level>Warning</Level>
        <Reason>FailedScheduling</Reason>
  </Data>
  <Code>0</Code>
</DescribeEventMonitorList>
```

`JSON` 格式

```
{
    "TotalCount": 12,
    "PageSize": 20,
    "RequestId": "A1274507-33EE-13DD-A1EB-9B55F43E2F9E",
    "PageNumber": 1,
    "Data": [
        {
            "PodName": "jck-deployment-yacs-24209-31780-748808-6d4cc4bc4-tbm4j",
            "Message": "Started container jck-app-container-24209-31780",
            "EventTime": "2021-07-28T09:16Z",
            "NameSpace": "jck-namespace-24209",
            "Level": "Normal",
            "HostName": "eci",
            "Reason": "Started"
        },
        {
            "PodName": "jck-deployment-yacs-24209-31780-748808-6d4cc4bc4-tbm4j",
            "Message": "0/2 nodes are available: 1 Insufficient cpu, 1 node(s) had
taint {virtual-kubelet.io/provider: alibabacloud}, that the pod didn't tolerate.",
            "EventTime": "2021-07-28T09:15Z",
            "NameSpace": "jck-namespace-24209",
            "Level": "Warning",
            "Reason": "FailedScheduling"
        }
    ],
    "Code": 0
}
```

错误码

HttpCode	错误码	错误信息	描述
500	system.error	A system error occurred.	系统异常

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

4.3. DescribeAppMonitorMetric

调用DescribeAppMonitorMetric查询云应用基础监控指标

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。

请求参数

名称	类型	是否必选	示例值	描述

Action	String	是	DescribeAppMonitorMetric	系统规定参数。取值：DescribeAppMonitorMetric。
AppId	Long	是	1123	云应用id
DeployOrderId	String	是	1222	发布单id
EndTime	Long	是	1627487322000	查询截止时间
EnvId	Long	是	1122	云应用环境id
Metric	String	是	pod.cpu.usage_rate	POD_MEM_WORKING_SET("pod.memory.working_set", "实例内存使用量"), POD_CPU_USAGE("pod.cpu.usage_rate", "实例CPU使用量"), POD_NETWORK_RX_RATE("pod.network.rx_rate", "实例网络接受速率"), POD_NETWORK_RX_ERRORS_RATE("pod.network.rx_errors_rate", "实例网络接收错误速率"), POD_CPU_UTILIZATION("pod.cpu.utilization", "实例CPU使用率"), POD_MEM_UTILIZATION("pod.memory.utilization", "实例内存使用率"), POD_NETWORK_TX_RATE("pod.network.tx_rate", "实例网络发送速率"), POD_NETWORK_TX_ERRORS_RATE("pod.network.tx_errors_rate", "实例网络接发送误速率"), GROUP_CPU_USAGE_RATE("deployment.cpu.usage_rate", "分组内存使用量"), GROUP_MEMORY_WORKING_SET("deployment.memory.working_set", "分组内存使用量"), GROUP_NETWORK_RX_RATE("deployment.network.rx_rate", "分组网络接受速率"), GROUP_NETWORK_RX_ERRORS_RATE("deployment.network.rx_errors_rate", "分组网络接收错误速率"), GROUP_NETWORK_TX_RATE("deployment.network.tx_rate", "分组网络发送速率"), GROUP_NETWORK_TX_ERRORS_RATE("deployment.network.tx_errors_rate", "分组网络接发送误速率"),

PodName	String	是	jck-deployment-yacs-24209-25073-728587-5757b8df59-rzpw6	POD实例名称
StartTime	Long	是	1627487322000	查询开始时间
Type	String	是	instance	查询类型，group | instance。如果为group，则查询发布单(分组)维度的指标，需要传入发布单id字段；如果为instance，则查询某个pod实例的指标，需要传入podName字段

返回数据

名称	类型	示例值	描述
Code	Integer	200	返回码
ErrMsg	String	xxx	错误信息
RequestId	String	xxx	请求id
Result	Array of MetricItem		结果
Categories	List	["2021-07-28 08:39:00"]	时间序列
Data	List	[0,0]	值序列
Name	String	CPU使用量	指标名称
Success	Boolean	true	是否正常

示例

请求示例

```
http(s)://[Endpoint]/?Action=DescribeAppMonitorMetric
&AppId=1123
&DeployOrderId=1222
&EndTime=1627487322000
&EnvId=1122
&Metric=pod.cpu.usage_rate
&PodName=jck-deployment-yacs-24209-25073-728587-5757b8df59-rzpw6
&StartTime=1627487322000
&Type=instance
&<公共请求参数>
```

正常返回示例

`XML` **格式**

```
<DescribeAppMonitorMetric>
  <RequestId>457582AE-9403-1FC6-B875-0BDB5336C357</RequestId>
  <Code>0</Code>
  <Success>true</Success>
  <Result>
        <Categories>2021-07-28 08:39:00</Categories>
        <Categories>2021-07-28 08:40:00</Categories>
        <Categories>2021-07-28 08:41:00</Categories>
        <Categories>2021-07-28 08:42:00</Categories>
        <Categories>2021-07-28 08:43:00</Categories>
        <Categories>2021-07-28 08:44:00</Categories>
        <Categories>2021-07-28 08:45:00</Categories>
        <Categories>2021-07-28 08:46:00</Categories>
        <Categories>2021-07-28 08:47:00</Categories>
        <Categories>2021-07-28 08:48:00</Categories>
        <Data>0.005</Data>
        <Data>0.006</Data>
        <Data>0.006</Data>
        <Data>0.005</Data>
        <Data>0.005</Data>
        <Data>0.006</Data>
        <Data>0.006</Data>
        <Data>0.006</Data>
        <Data>0.006</Data>
        <Data>0.006</Data>
        <Name>实例CPU使用量</Name>
  </Result>
</DescribeAppMonitorMetric>
```

`JSON` **格式**

```
{
    "RequestId": "457582AE-9403-1FC6-B875-0BDB5336C357",
    "Code": 0,
    "Success": true,
    "Result": [
        {
            "Categories": [
                "2021-07-28 08:39:00",
                "2021-07-28 08:40:00",
                "2021-07-28 08:41:00",
                "2021-07-28 08:42:00",
                "2021-07-28 08:43:00",
                "2021-07-28 08:44:00",
                "2021-07-28 08:45:00",
                "2021-07-28 08:46:00",
                "2021-07-28 08:47:00",
                "2021-07-28 08:48:00"
            ],
            "Data": [
                0.005,
                0.006,
                0.006,
                0.005,
                0.005,
                0.006,
                0.006,
                0.006,
                0.006,
                0.006
            ],
            "Name": "实例CPU使用量"
        }
    ]
}
```


错误码

HttpCode	错误码	错误信息	描述
500	system.error	A system error occurred.	系统异常

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

5.应用发布

5.1. 发布

5.1.1. DeployApp

调用DeployApp发布部署应用。按环境来发布应用，将应用按照环境配置上指定的实例数量以及部署配置部署至集群。

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	DeployApp	系统规定参数。取值：DeployApp。
EnvId	Long	是	123	环境id。
Name	String	是	发布单名称	发布单的名称。
TotalPartitions	Integer	是	2	发布划分的批次数。 建议至少划分为2个批次，划分多个批次的好处主要有两点： 1、做金丝雀灰度，防止本次变更有问题而导致服务整体宕机；建议在第一批次发布完成后暂停观察一段时间，若通过系统监控或查看日志发现变更有问题，则可以停止发布并回滚； 2、分批重启应用实例，防止发布期间所有实例同时重启而导致服务不可用；
DeployPacketUrl	String	否	https://myoss.oss-cn-zhangjiakou.aliyuncs.com/webDemo.jar	代码包下载地址。 如果需要系统下载代码包，提供下载代码包的HTTP或HTTPS地址
DeployPacketId	Long	否	123	代码包id。 如果需要系统下载代码包且使用聚石塔代码包管理模块管理代码包，此处传递聚石塔提供的代码包id
Description	String	否	第一个版本发布	发布单的描述。

PauseType	String	否	first	分批发布暂停方式。取值范围： first：第一批暂停；第一批发布完成后暂停发布； each：每批暂停；每个批次发布完成后都暂停发布； none：不暂停；每个批次发布完成后自动发布下一批次，中间不暂停； 默认为none。建议使用first或each，可有效降低一次变更有问题而导致服务不可用的风险。
ArmsFlag	Boolean	否	true	是否开通arms监控。取值范围： true：开通 false：不开通 默认为false
UpdateStrategyType	String	否	offline_old_then_online_new	应用实例更新策略（仅支持无状态应用设置此策略）： offline_old_then_online_new：先下线旧版本再上线新版本，默认为此策略 online_new_then_offline_old：先上线新版本再下线旧版本
ContainerImageList.N	RepeatList	否	registry-vpc.cn-zhangjiakou.aliyuncs.com/XXXX/XXXX:1.0	与yaml配置中containers配置下的镜像**个数、顺序一致**的镜像列表，若不传，则使用原配置中的镜像版本，若传此列表则使用传入的镜像版本发布。
InitContainerImageList.N	RepeatList	否	registry-vpc.cn-zhangjiakou.aliyuncs.com/XXXX/XXXX:1.0	与yaml配置中initContainers配置下的镜像**个数、顺序一致**的镜像列表，若不传，则使用原配置中的镜像版本，若传此列表则使用传入的镜像版本发布。

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功。
ErrMsg	String	this is a tip message	错误信息。
RequestId	String	DEFBA135-1B1E-4B15-A514-6D7E9BAD7EDC	请求id。

Result	Struct		结果。
Admitted	Boolean	true	发布是否准入。取值范围： true：准许发布； false：禁止发布；如果处于封网阶段，则为false
BusinessCode	String	102000	具体的错误码。枚举值： 102000：没有发布准入凭据； 102001：发布准入凭据审核中
DeployOrderId	Long	12029	发布单id。 发布单的唯一id，查询及控制发布单时需要传递
Success	Boolean	true	是否成功。

示例

请求示例

```
http(s)://[Endpoint]/?Action=DeployApp
&EnvId=123
&Name=发布单名称
&TotalPartitions=2
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<DeployAppResponse>
      <RequestId>DEFBA135-1B1E-4B15-A514-6D7E9BAD7EDC</RequestId>
      <Code>0</Code>
      <Success>true</Success>
      <Result>
            <Admitted>true</Admitted>
            <DeployOrderId>12029</DeployOrderId>
      </Result>
</DeployAppResponse>
```

`JSON` 格式

```
{
  "RequestId": "DEFBA135-1B1E-4B15-A514-6D7E9BAD7EDC",
  "Code": 0,
  "Success": true,
  "Result": {
    "Admitted": true,
    "DeployOrderId": 12029
  }
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

5.1.2. ListDeployOrders

查询发布单列表。根据用户指定的查询条件分页查询发布单列表。

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	ListDeployOrders	系统规定参数。取值：ListDeployOrders。
PageNumber	Integer	是	1	页码。
PageSize	Integer	是	20	分页大小，每页的记录数量。
AppId	Long	是	123	应用id。
EnvId	Long	是	123	环境id。
EnvType	String	是	online	环境类型。
DeployType	String	是	partition_parallel	发布方式。目前默认是分批发布，不需要传递。

PauseType	String	是	first	分批发布暂停方式。枚举值： first：第一批暂停； none：不暂停； each：每批暂停；
DeployCategory	String	是	deploy	发布单分类。枚举值： deploy：发布部署； scale：扩缩容；
PartitionType	String	是	group_even	分批方式。当前是默认分批方式，不需要传递。
EndTimeLessThan	String	是	2020-01-01 00:00:00	发布结束时间早于。
EndTimeLessThanOrEqualTo	String	是	2020-01-01 00:00:00	发布结束时间早于或等于。
EndTimeGreaterThan	String	是	2020-01-01 00:00:00	发布结束时间晚于。
EndTimeGreaterThanOrEqualTo	String	是	2020-01-01 00:00:00	发布结束时间晚于或等于。
StartTimeGreaterThan	String	是	2020-01-01 00:00:00	发布开始时间晚于。
StartTimeGreaterThanOrEqualTo	String	是	ç	发布开始时间晚于或等于。
StartTimeLessThan	String	是	2020-01-01 00:00:00	发布开始时间早于。
StartTimeLessThanOrEqualTo	String	是	2020-01-01 00:00:00	发布开始时间早于或等于。
Status	Integer	是	3	发布单状态。枚举值： 1：部署中； 2：暂停； 3：已完成；
ResultList.N	RepeatList	否	1	发布结果。枚举值： 0：部署中； 1：成功； 2：失败； 4：人为置为失败（如手工关闭发布单）；

StatusList.N	RepeatList	否	3	发布单状态。枚举值： 1：部署中； 2：暂停； 3：已完成；

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功。
ErrorMsg	String	this is a tip message	错误信息。
PageNumber	Integer	1	页码。
PageSize	Integer	10	分页大小。每页的记录数量。
RequestId	String	FD0E1170-xxxx-xxxx-8D2F-0E32F904F169	请求id。
TotalCount	Long	95	记录总数量。
Data	Array		分页数据。
AppInstanceType	String	k8s_pod	应用实例类型。容器实例为k8s_pod。
CurrentPartitionNum	Integer	1	当前批次号。分批发布时处于发布中的批次。
DeployOrderId	Long	123	发布单id。
DeployPauseType	String	first	发布暂停方式。
DeployPauseTypeName	String	第一批暂停	发布暂停方式显示名称。
DeployType	String	partition_parallel	发布方式。
DeployTypeName	String	分批发布	发布方式显示名称。
Description	String	发布单描述	发布单描述。

ElapsedTime	Integer	20	发布耗时，单位是秒。
EndTime	String	2020-01-01 00:00:00	发布结束时间。
EnvId	Long	123	环境id。
EnvType	String	online	环境类型。
FailureRate	String	0	部署失败率。
FinishAppInstanceCt	Integer	2	完成部署的实例数量。
Name	String	发布单名称	发布单名称
PartitionType	String	group_even	分批方式。
PartitionTypeName	String	分组均分	分批方式显示名称。
Result	Integer	1	发布结果。
ResultName	String	成功	发布结果显示名称。
SchemaId	Long	123	发布单对应环境的部署配置id。
StartTime	String	2020-01-01 00:00:00	发布开始时间。
Status	Integer	3	发布单状态。
StatusName	String	已完成	发布单状态显示名称。
TotalAppInstanceCt	Integer	2	发布单总的应用实例数量。
TotalPartitions	Integer	2	分批数量。
UserId	String	123	用户id。
UserNick	String	测试用户	用户昵称。

示例

请求示例

```
http(s)://[Endpoint]/?Action=ListDeployOrders
&PageNumber=1
&PageSize=20
&AppId=123
&EnvId=123
&EnvType=online
&DeployType=partition_parallel
&PauseType=first
&DeployCategory=deploy
&PartitionType=group_even
&EndTimeLessThan=2020-01-01 00:00:00
&EndTimeLessThanOrEqualTo=2020-01-01 00:00:00
&EndTimeGreaterThan=2020-01-01 00:00:00
&EndTimeGreaterThanOrEqualTo=2020-01-01 00:00:00
&StartTimeGreaterThan=2020-01-01 00:00:00
&StartTimeGreaterThanOrEqualTo=2020-01-01 00:00:00
&StartTimeLessThan=2020-01-01 00:00:00
&StartTimeLessThanOrEqualTo=2020-01-01 00:00:00
&Status=3
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<ListDeployOrdersResponse>
      <TotalCount>95</TotalCount>
      <PageSize>10</PageSize>
      <RequestId>847A742C-9A6A-4D37-B0FD-D26D4F32F07E</RequestId>
      <PageNumber>1</PageNumber>
      <Data>
            <DeployType>partition_parallel</DeployType>
            <Status>3</Status>
            <TotalPartitions>1</TotalPartitions>
            <EndTime>2020-01-05T17:03Z</EndTime>
            <DeployPauseType>none</DeployPauseType>
            <EnvId>123</EnvId>
            <StartTime>2020-01-05T17:03Z</StartTime>
            <ElapsedTime>1</ElapsedTime>
            <StatusName>已完成</StatusName>
            <DeployTypeName>分批发布-批内并行</DeployTypeName>
            <PartitionType>group_even</PartitionType>
            <EnvType>online</EnvType>
            <DeployOrderId>123</DeployOrderId>
            <Name>应用实例数扩缩容至1个</Name>
            <PartitionTypeName>分组均分</PartitionTypeName>
            <Result>1</Result>
            <ResultName>成功</ResultName>
            <AppInstanceType>k8s_pod</AppInstanceType>
            <UserId>123</UserId>
            <DeployPauseTypeName>不暂停</DeployPauseTypeName>
            <UserNick>测试</UserNick>
      </Data>
      <Code>0</Code>
</ListDeployOrdersResponse>
```

`JSON` **格式**

```
{
  "TotalCount": 95,
  "PageSize": 10,
  "RequestId": "847A742C-9A6A-4D37-B0FD-D26D4F32F07E",
  "PageNumber": 1,
  "Data": [
    {
      "DeployType": "partition_parallel",
      "Status": 3,
      "TotalPartitions": 1,
      "EndTime": "2020-01-05T17:03Z",
      "DeployPauseType": "none",
      "EnvId": 123,
      "StartTime": "2020-01-05T17:03Z",
      "ElapsedTime": 1,
      "StatusName": "已完成",
      "DeployTypeName": "分批发布-批内并行",
      "PartitionType": "group_even",
      "EnvType": "online",
      "DeployOrderId": 123,
      "Name": "应用实例数扩缩容至1个",
      "PartitionTypeName": "分组均分",
      "Result": 1,
      "ResultName": "成功",
      "AppInstanceType": "k8s_pod",
      "UserId": "123",
      "DeployPauseTypeName": "不暂停",
      "UserNick": "测试"
    }
  ],
  "Code": 0
}
```


错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

5.1.3. 应用扩缩容

调用ScaleApp扩缩容应用。按照环境维度将应用扩缩容至用户指定的副本数量。

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	ScaleApp	系统规定参数。取值：**ScaleApp**。

EnvId	Long	是	123	环境id。
Replicas	Integer	是	1	应用实例数量。
TotalPartitions	Integer	否	1	扩缩容划分的批次数，不指定默认为2批。

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功。
ErrMsg	String	t his is a t ip message	错误信息。
RequestId	String	xxxx-xxxx-xxxx-xxxx	请求id
Success	Boolean	true	是否成功。
Result	Object		结果。
BusinessCode	String	0	具体的错误码。枚举值: </br> 102000:没有发布准入凭据; </br> 102001:发布准入凭据审核中
DeployOrderId	Long	123456	发布单id。 </br> 发布单的唯一id，查询及控制发布单时需要传递
Admitted	Boolean	true	发布是否准入。取值范围: </br> t rue:准许发布;</br> f alse:禁止发布;如果处于封网阶段，则为 false

示例

请求示例

```
http(s)://[Endpoint]/?Action=ScaleApp
&EnvId=123
&Replicas=1
&TotalPartitions=1
&公共请求参数
```

正常返回示例

`XML` 格式

```
HTTP/1.1 200 OK
Content-Type:application/xml

<ScaleAppResponse>
    <Code>0</Code>
    <ErrMsg>t his is a t ip message</ErrMsg>
    <Success>true</Success>
    <Result>
        <BusinessCode>0</BusinessCode>
        <DeployOrderId>123456</DeployOrderId>
        <Admitted>true</Admitted>
    </Result>
</ScaleAppResponse>
```

`JSON` 格式

```
HTTP/1.1 200 OK
Content-Type:application/json

{
  "Code" : 0,
  "ErrMsg" : "t his is a t ip message",
  "Success" : true,
  "Result" : {
    "BusinessCode" : "0",
    "DeployOrderId" : 123456,
    "Admitted" : true
  }
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

5.1.4. DescribeDeployOrderDetail

调用DescribeDeployOrderDetail查询发布单详情。查询一个发布单的详细信息。

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。

请求参数

名称	类型	是否必选	示例值	描述

Action	String	是	DescribeDeployOrderDetail	系统规定参数。取值：DescribeDeployOrderDetail。
DeployOrderId	Long	是	123	发布单id。

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功。
ErrMsg	String	this is a tip message	错误信息。
RequestId	String	FD0E1170-xxxx-xxxx-8D2F-0E32F904F169	请求id。
Result	Struct		结果。
AppInstanceType	String	k8s_pod	应用实例类型。容器实例为k8s_pod。
CurrentPartitionNum	Integer	1	当前批次号。分批发布时处于发布中的批次。
DeployOrderId	Long	123	发布单id。
DeployPauseType	String	first	发布暂停方式。
DeployPauseTypeName	String	第一批暂停	发布暂停方式显示名称。
DeployType	String	partition_parallel	发布方式。
DeployTypeName	String	分批发布	发布方式显示名称。
Description	String	发布单描述	发布单描述。
ElapsedTime	Integer	20	发布耗时，单位是秒。
EndTime	String	2020-01-01 00:00:00	发布结束时间。

EnvId	Long	123	环境id。
EnvType	String	online	环境类型。
FailureRate	String	0	部署失败率。
FinishAppInstanceCt	Integer	1	完成部署的实例数量。
Name	String	发布单名称	发布单名称。
PartitionType	String	group_even	分批方式。
PartitionTypeName	String	分组均分	分批方式显示名称。
Result	Integer	1	发布结果。 0：待确认 1：成功 其他数值：失败
ResultName	String	成功	发布结果显示名称。
SchemaId	Long	123	发布单对应环境的部署配置id。
StartTime	String	2020-01-01 00:00:00	发布开始时间。
Status	Integer	3	发布单状态。 0：等待部署（处于排队状态等） 1：部署中 2：暂停 3：已完成
StatusName	String	已完成	发布单状态显示名称。
TotalAppInstanceCt	Integer	2	发布单总的应用实例数量。
TotalPartitions	Integer	2	分批数量。
UserId	String	123	用户id。

UserNick	String	测试用户	用户昵称。
Success	Boolean	true	是否成功。

示例

请求示例

```
http(s)://[Endpoint]/?Action=DescribeDeployOrderDetail
&DeployOrderId=123
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<DescribeDeployOrderDetailResponse>
      <RequestId>3711002F-9DB0-461E-BE1B-E1ABD8F6A348</RequestId>
      <Code>0</Code>
      <Success>true</Success>
      <Result>
            <DeployType>partition_parallel</DeployType>
            <TotalPartitions>1</TotalPartitions>
            <EndTime>2019-12-31T14:25Z</EndTime>
            <DeployPauseType>first</DeployPauseType>
            <EnvId>123</EnvId>
            <ElapsedTime>31</ElapsedTime>
            <DeployTypeName>分批发布-批内并行</DeployTypeName>
            <FailureRate>0</FailureRate>
            <EnvType>online</EnvType>
            <Result>1</Result>
            <DeployOrderId>123</DeployOrderId>
            <Name>lingfeng1230-1</Name>
            <TotalAppInstanceCt>1</TotalAppInstanceCt>
            <ResultName>成功</ResultName>
            <CurrentPartitionNum>1</CurrentPartitionNum>
            <DeployPauseTypeName>第一批暂停</DeployPauseTypeName>
            <UserNick>测试66</UserNick>
            <Status>3</Status>
            <StartTime>2019-12-31T14:25Z</StartTime>
            <StatusName>已完成</StatusName>
            <PartitionType>group_even</PartitionType>
            <PartitionTypeName>分组均分</PartitionTypeName>
            <AppInstanceType>k8s_pod</AppInstanceType>
            <UserId>123</UserId>
            <SchemaId>123</SchemaId>
            <FinishAppInstanceCt>0</FinishAppInstanceCt>
      </Result>
</DescribeDeployOrderDetailResponse>
```

`JSON` 格式

```
{
  "RequestId": "3711002F-9DB0-461E-BE1B-E1ABD8F6A348",
  "Code": 0,
  "Success": true,
  "Result": {
    "DeployType": "partition_parallel",
    "TotalPartitions": 1,
    "EndTime": "2019-12-31T14:25Z",
    "DeployPauseType": "first",
    "EnvId": 123,
    "ElapsedTime": 31,
    "DeployTypeName": "分批发布-批内并行",
    "FailureRate": 0,
    "EnvType": "online",
    "Result": 1,
    "DeployOrderId": 123,
    "Name": "lingfeng1230-1",
    "TotalAppInstanceCt": 1,
    "ResultName": "成功",
    "CurrentPartitionNum": 1,
    "DeployPauseTypeName": "第一批暂停",
    "UserNick": "测试66",
    "Status": 3,
    "StartTime": "2019-12-31T14:25Z",
    "StatusName": "已完成",
    "PartitionType": "group_even",
    "PartitionTypeName": "分组均分",
    "AppInstanceType": "k8s_pod",
    "UserId": "123",
    "SchemaId": 123,
    "FinishAppInstanceCt": 0
  }
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

5.1.5. ResumeDeploy

调用ResumeDeploy恢复并继续发布。对于处于暂停状态（如DeployApp时指定的PauseType为first或each等）的发布单，恢复至发布状态。

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。

请求参数

名称	类型	是否必选	示例值	描述

Action	String	是	ResumeDeploy	系统规定参数。取值：ResumeDeploy。
DeployOrderId	Long	是	123	发布单id。

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功。
ErrMsg	String	this is a tip message	错误信息。
RequestId	String	FD0E1170-xxxx-xxxx-8D2F-0E32F904F169	请求id。
Success	Boolean	true	是否成功。

示例

请求示例

```
http(s)://[Endpoint]/?Action=ResumeDeploy
&DeployOrderId=123
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<ResumeDeployResponse>
      <RequestId>69FB121A-CD59-4046-ABD0-459C3F51D60E</RequestId>
      <Code>0</Code>
      <Success>true</Success>
</ResumeDeployResponse>
```

`JSON` 格式

```
{
  "RequestId": "69FB121A-CD59-4046-ABD0-459C3F51D60E",
  "Code": 0,
  "Success": true
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

5.1.6. CloseDeployOrder

调用CloseDeployOrder关闭发布单。关闭一个处于【发布中】状态的发布单，系统会将发布单状态置为【已完成】，但同时将发布结果置为【失败】。对于有问题的发布单，一定要将其关闭，否则系统判断存在未完成的发布单则不允许进行新的发布。

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	CloseDeployOrder	系统规定参数。取值：CloseDeployOrder。
DeployOrderId	Long	是	123	发布单id。

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功。
ErrMsg	String	this is a tip message	错误信息。
RequestId	String	FD0E1170-xxxx-xxxx-8D2F-0E32F904F169	请求id。
Success	Boolean	true	是否成功。

示例

请求示例

```
http(s)://[Endpoint]/?Action=CloseDeployOrder
&DeployOrderId=123
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<CloseDeployOrderResponse>
      <RequestId>69FB121A-CD59-4046-ABD0-459C3F51D60E</RequestId>
      <Code>0</Code>
      <Success>true</Success>
</CloseDeployOrderResponse>
```

`JSON` 格式

```
{
  "RequestId": "69FB121A-CD59-4046-ABD0-459C3F51D60E",
  "Code": 0,
  "Success": true
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

5.2. 部署实例(POD)查询

5.2.1. ListPods

调用获取环境下所有的pod列表。根据用户指定的查询条件分页查询POD列表。

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	ListPods	系统规定参数。取值：ListPods。
DeployOrderId	Long	是	123	发布单id。
PageNumber	Integer	是	1	页码。
PageSize	Integer	是	10	分页大小，每页的记录数量。
ResultList.N	RepeatList	否	1	部署结果。枚举值： 0：部署中； 1：成功； 2：失败； 4：人为置为失败（如手工关闭发布单）；
StatusList.N	RepeatList	否	3	部署状态。枚举值： 1：部署中； 2：暂停； 3：已完成；

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功。
ErrorMsg	String	this is a tip message	错误信息。
PageNumber	Integer	1	页码。
PageSize	Integer	10	分页大小。每页的记录数量。
RequestId	String	FD0E1170-xxxx-xxxx-8D2F-0E32F904F169	请求id。
TotalCount	Long	12	记录总数量。
Data	Array		分页数据。
AppInstanceId	String	xxx-xx	应用实例id。
DeployOrderId	Long	123	发布单id。
DeployPartitionNum	Integer	1	发布分批号。
GroupName	String	defaultGroup	分组名称。
HostIp	String	127.2.1.1	主机ip。
HostName	String	xxx	主机名称。
PodIp	String	127.2.1.1	POD IP。
Region	String	cn-zhangjiakou	Region
Result	Integer	1	部署结果。
ResultName	String	成功	部署结果名称。
StartTime	String	2020-01-01 00:00:00	部署开始时间。

Status	Integer	3	应用实例部署状态。
StatusName	String	已完成	应用实例部署状态显示名称。
UpdateTime	String	2020-01-01 00:00:00	更新时间。
DeploySteps	Array		部署步骤。
Status	String	SUCCESS	部署步骤状态
StepCode	String	StartAppInstance	部署步骤Code。
StepName	String	启动应用	部署步骤名称。

示例

请求示例

```
http(s)://[Endpoint]/?Action=ListPods
&DeployOrderId=123
&PageNumber=1
&PageSize=10
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<ListPodsResponse>
  <TotalCount>0</TotalCount>
  <PageSize>10</PageSize>
  <RequestId>0CDC3CE4-CF42-4FF1-B5D3-3AED18E1350D</RequestId>
  <PageNumber>1</PageNumber>
  <Data>
        <GroupName>lingfeng-xxx-host</GroupName>
        <HostIp>192.168.17.206</HostIp>
        <Status>3</Status>
        <ResultName>成功</ResultName>
        <PodIp>172.20.3.49</PodIp>
        <UpdateTime>2020-01-05T18:38:49</UpdateTime>
        <StatusName>已完成</StatusName>
        <DeploySteps>
              <Status>SUCCESS</Status>
              <StepCode>StartAppInstance</StepCode>
              <StepName>启动应用</StepName>
        </DeploySteps>
        <DeployOrderId>123</DeployOrderId>
        <Result>1</Result>
  </Data>
  <Code>0</Code>
</ListPodsResponse>
```

`JSON` **格式**

```
{
  "TotalCount": 0,
  "PageSize": 10,
  "RequestId": "0CDC3CE4-CF42-4FF1-B5D3-3AED18E1350D",
  "PageNumber": 1,
  "Data": [
    {
      "GroupName": "lingfeng-xxx-host",
      "HostIp": "192.168.17.206",
      "Status": 3,
      "ResultName": "成功",
      "PodIp": "172.20.3.49",
      "UpdateTime": "2020-01-05T18:38:49",
      "StatusName": "已完成",
      "DeploySteps": [
        {
          "Status": "SUCCESS",
          "StepCode": "StartAppInstance",
          "StepName": "启动应用"
        }
      ],
      "DeployOrderId": 123,
      "Result": 1
    }
  ],
  "Code": 0
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

5.2.2. DescribePodLog

调用DescribePodLog查询pod启动日志。启动日志中包含代码下载日志及容器内应用标准输出日志。

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	DescribePodLog	系统规定参数。取值：DescribePodLog。
AppInstId	String	是	xx	应用实例id

DeployOrderId	Long	是	123	部署单id

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功
ErrMsg	String	this is a tip message	错误信息
RequestId	String	FD0E1170-xxxx-xxxx-8D2F-0E32F904F169	请求id
Success	Boolean	true	是否成功。
Result	Struct		结果。
DeployOrderName	String	xx	部署单名称
EnvTypeName	String	online	环境类型名称，online 或 test
DeployStepList	Array		部署步骤列表
StepName	String	启动应用	步骤名称：启动应用、停止应用、删除应用
StepCode	String	StartAppInstance	步骤代码，StartAppInstance、StopAppInstance、DeleteAppInstance
StepLog	String	xxxxx	日志
Status	String	SUCCESS	当前步骤的状态，处理中: PROCESSING, 成功: SUCCESS, 失败: FAIL

示例

请求示例

```
http(s)://[Endpoint]/?Action=DescribePodLog
&AppInstId=xx
&DeployOrderId=123
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<DescribePodLogResponse>
      <RequestId>49799724-FBC6-48D9-A82A-375D6B3AC060</RequestId>
      <Code>0</Code>
      <Success>true</Success>
      <Result>
            <DeployOrderName>lingfeng1230-1</DeployOrderName>
            <EnvTypeName>正式</EnvTypeName>
            <DeployStepList>
                  <Status>SUCCESS</Status>
                  <StepCode>StartAppInstance</StepCode>
                  <StepLog>xxx</StepLog>
                  <StepName>启动应用</StepName>
            </DeployStepList>
      </Result>
</DescribePodLogResponse>
```

`JSON` 格式

```
{
  "RequestId": "49799724-FBC6-48D9-A82A-375D6B3AC060",
  "Code": 0,
  "Success": true,
  "Result": {
    "DeployOrderName": "lingfeng1230-1",
    "EnvTypeName": "正式",
    "DeployStepList": [
      {
        "Status": "SUCCESS",
        "StepCode": "StartAppInstance",
        "StepLog": "xxx",
        "StepName": "启动应用"
      }
    ]
  }
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

5.2.3. DescribePodEvents

调用DescribePodEvents查询POD发布事件。事件的状态会随着时间被集群清理，建议在发布过程中查询。

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	DescribePodEvents	系统规定参数。取值：DescribePodEvents。
AppInstId	String	是	xxx-xx	应用实例id。
DeployOrderId	Long	是	123	发布单id。

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功。
ErrMsg	String	this is a tip message	错误信息。
RequestId	String	FD0E1170-xxxx-xxxx-8D2F-0E32F904F169	请求id。
Result	Struct		结果。
DeployOrderName	String	xxx	发布单名称。
PodEvents	Array		POD Events
Action	String	xxxx	Action
Count	Integer	1	数量。
EventTime	String	2020-01-05T18:38Z	事件发生时间。
FirstTimestamp	String	2020-01-05T18:38Z	首次时间。
LastTimestamp	String	2020-01-05T18:38Z	最后一次时间。
Message	String	Successfully assigned	事件消息。
Reason	String	Scheduled	事件原因。

Type	String	Normal	事件类型。
Success	Boolean	true	是否成功。

示例

请求示例

```
http(s)://[Endpoint]/?Action=DescribePodEvents
&AppInstId=xxx-xx
&DeployOrderId=123
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<DescribePodEventsResponse>
      <RequestId>8BE0B4A0-12B7-485E-95B4-618E2BBD6438</RequestId>
      <Code>0</Code>
      <Success>true</Success>
      <Result>
            <DeployOrderName>xxxx</DeployOrderName>
            <PodEvents>
                  <Type>Normal</Type>
                  <LastTimestamp>2020-01-05T18:38Z</LastTimestamp>
                  <Message>Successfully assigned</Message>
                  <Count>1</Count>
                  <FirstTimestamp>2020-01-05T18:38Z</FirstTimestamp>
                  <Reason>Scheduled</Reason>
            </PodEvents>
      </Result>
</DescribePodEventsResponse>
```

`JSON` 格式

```
{
  "RequestId": "8BE0B4A0-12B7-485E-95B4-618E2BBD6438",
  "Code": 0,
  "Success": true,
  "Result": {
    "DeployOrderName": "xxxx",
    "PodEvents": [
      {
        "Type": "Normal",
        "LastTimestamp": "2020-01-05T18:38Z",
        "Message": "Successfully assigned",
        "Count": 1,
        "FirstTimestamp": "2020-01-05T18:38Z",
        "Reason": "Scheduled"
      }
    ]
  }
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

6.集群运维

6.1. 存储卷管理

6.1.1. CreatePersistentVolume

调用CreatePersistentVolume在集群下创建一个持久化存储卷

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	CreatePersistentVolume	系统规定参数。取值：**CreatePersistentVolume**。
ClusterInstanceId	String	是	xxx	集群实例id
Name	String	是	test-pv	持久化存储卷名称，不可与已有重复，名称必须以小写字母开头，只能包含小写字母、数字、"."和"-"
Capacity	String	是	500Gi	申请持久化存储卷的容量大小，支持纯数字、定点整数+以下后缀：E, P, T, G, M, K 、2的幂形式：Ei, Pi, Ti, Gi, Mi, Ki
AccessModes	String	是	ReadWriteMany	持久化存储卷的访问模式，可取值：ReadWriteMany、ReadWriteOnce
MountTargetDomain	String	是	xxxxxx.xxxxx.cn-zhangjiakou.nas.aliyuncs.com	nas挂载点的域名
MountDir	String	否	/data	挂载到nas的目录，如：/xxx，默认为根目录
ReclaimPolicy	String	是	Retain	存储卷回收策略，支持以下值：Retain、Delete
NFSVersion	String	是	4.0	NFS挂载协议的版本号，支持3和4.0

StorageClass	String	是	test-pv-class	存储类名称，不能与已有重复，名称必须以小写字母开头，只能包含小写字母、数字、"."和"-"
NasType	String	否	NFS	可取值 • NFS: 一般Linux文件系统使用； • SMB: 一般windows文件系统使用； 不确定请留空

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功
RequestId	String	xx	请求id
ErrMsg	String	参数错误	错误信息
Result	Object		返回结果
PersistentVolumeId	Long	0	持久化存储卷id

示例

请求示例

```
http(s)://[Endpoint]/?Action=CreatePersistentVolume
&ClusterInstanceId=xxx
&Name=test-pv
&Capacity=500Gi
&AccessModes=ReadWriteMany
&MountTargetDomain=xxxxxx.xxxxx.cn-zhangjiakou.nas.aliyuncs.com
&MountDir=/data
&ReclaimPolicy=Retain
&NFSVersion=4.0
&StorageClass=test-pv-class
&NasType=NFS
&公共请求参数
```

正常返回示例

`XML` 格式

```
HTTP/1.1 200 OK
Content-Type:application/xml

<CreatePersistentVolumeResponse>
    <Result>
        <PersistentVolumeId>0</PersistentVolumeId>
    </Result>
    <RequestId>xxx-4748-4442-A2DA-16C419E52E71</RequestId>
    <Code>0</Code>
</CreatePersistentVolumeResponse>
```

`JSON` 格式

```
HTTP/1.1 200 OK
Content-Type:application/json

{
  "Result" : {
    "PersistentVolumeId" : 0
  },
  "RequestId" : "xxx-4748-4442-A2DA-16C419E52E71",
  "Code" : 0
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

6.1.2. ListPersistentVolume

调用ListPersistentVolume获取持久化存储卷列表

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	ListPersistentVolume	系统规定参数。取值：ListPersistentVolume。
ClusterInstanceId	String	是	xxxx	集群实例id
PageNumber	Integer	是	1	页数
PageSize	Integer	是	20	每页大小，最大100

返回数据

名称	类型	示例值	描述
RequestId	String	xx	请求id
Code	Integer	0	返回码，0表示成功
PageSize	Integer	20	每页大小
PageNumber	Integer	1	页数
TotalCount	Long	18	总数
ErrMsg	String	xx	错误信息
Data	Array		返回数据
Name	String	xx	名称
Capacity	String	500Gi	容量，合法的格式：纯数字、定点整数+以下后缀：E, P, T, G, M, K 、2的幂形式：Ei, Pi, Ti, Gi, Mi, Ki。单位是bytes，字节。例如以下表示基本相等：128974848, 129e6, 129M, 123Mi
AccessModes	String	ReadWriteMany	访问模式，支持ReadWriteOnce：该卷可以被单个节点以读/写模式挂载、ReadWriteMany：该卷可以被多个节点以读/写模式挂载
ReclaimPolicy	String	Retain	回收策略，Retain（保留）：手动回收 Delete（删除）：关联的存储资产
Status	String	Available	状态，卷可以处于以下的某种状态： Available（可用）：一块空闲资源还没有被任何声明绑定 Bound（已绑定）：卷已经被声明绑定 Released（已释放）：声明被删除，但是资源还未被集群重新声明 Failed（失败）：该卷的自动回收失败
PvcName	String	xx	绑定的pvc名称，若未绑定此项为空

MountDir	String	/data	挂载目录
StorageClass	String	xxx	类，PV 可以具有一个类，通过将 storageClassName 属性设置为 StorageClass 的名称来指定该类。一个特定类别的 PV 只能绑定到请求该类别的 PVC。没有 storageClassName 的 PV 就没有类，它只能绑定到不需要特定类的 PVC。
Reason	String	xxx	创建失败的原因
CreateTime	String	2019-12-31 11:11:11	创建时间

示例

请求示例

```
http(s)://[Endpoint]/?Action=ListPersistentVolume
&ClusterInstanceId=xxxx
&PageNumber=1
&PageSize=20
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<ListPersistentVolumeResponse>
      <PageNumber>1</PageNumber>
      <Data>
            <Name>xx-xx-pv</Name>
            <Status>Bound</Status>
            <StorageClass>class1</StorageClass>
            <ReclaimPolicy>Retain</ReclaimPolicy>
            <CreateTime>2019-12-18 17:38:11</CreateTime>
            <Capacity>500Gi</Capacity>
            <AccessModes>ReadWriteMany</AccessModes>
      </Data>
      <TotalCount>6</TotalCount>
      <PageSize>1</PageSize>
      <RequestId>xx-843A-4EE9-867D-1C7054ED84C3</RequestId>
      <Code>0</Code>
</ListPersistentVolumeResponse>
```

`JSON` 格式

```
{
  "PageNumber": 1,
  "Data": [
    {
      "Name": "xx-xx-pv",
      "Status": "Bound",
      "StorageClass": "class1",
      "ReclaimPolicy": "Retain",
      "CreateTime": "2019-12-18 17:38:11",
      "Capacity": "500Gi",
      "AccessModes": "ReadWriteMany"
    }
  ],
  "TotalCount": 6,
  "PageSize": 1,
  "RequestId": "xx-843A-4EE9-867D-1C7054ED84C3",
  "Code": 0
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

6.1.3. DeletePersistentVolume

调用DeletePersistentVolume删除一个持久化存储卷，删除的持久化存储卷需是非Bound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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	DeletePersistentVolume	系统规定参数。取值：DeletePersistentVolume。
ClusterInstanceId	String	是	xxxxx	集群实例id
PersistentVolumeName	String	是	test-pv	持久化存储卷名称

返回数据

名称	类型	示例值	描述
RequestId	String	xx	请求id

Code	Integer	0	返回码，0表示成功
ErrMsg	String	xx	错误信息
Result	Struct		返回结果
Success	Boolean	true	是否成功

示例

请求示例

```
http(s)://[Endpoint]/?Action=DeletePersistentVolume
&ClusterInstanceId=xxxxx
&PersistentVolumeName=test-pv
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<DeletePersistentVolumeResponse>
      <Result>
            <Success>true</Success>
      </Result>
      <RequestId>xx-EA82-4BD1-98B7-EFE8FDA1523E</RequestId>
      <Code>0</Code>
</DeletePersistentVolumeResponse>
```

`JSON` 格式

```
{
  "Result": {
    "Success": true
  },
  "RequestId": "xx-EA82-4BD1-98B7-EFE8FDA1523E",
  "Code": 0
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

6.2. 节点资源打标

6.2.1. CreateNodeLabel

调用CreateNodeLabel创建节点标签。该接口用于创建标签而非绑定，绑定请使用接口BindNodeLabel

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	CreateNodeLabel	系统规定参数。取值：CreateNodeLabel。
ClusterId	String	是	abcd12345	集群实例ID
LabelKey	String	是	label.jst.com/type	标签Key值。 需要以字符串"label.jst.com/"开头。
LabelValue	String	是	web	标签value

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0为成功
ErrMsg	String	权限错误	错误日志
RequestId	String	1234ffadsf	请求ID
Success	Boolean	true	是否成功
Result	Struct		返回结果
ClusterId	String	iuysfb8381	集群ID
Id	Long	1	记录ID
LabelKey	String	label.jst.com/key	标签Key
LabelValue	String	value	标签值

示例

请求示例

```
http(s)://[Endpoint]/?Action=CreateNodeLabel
&ClusterId=abcd12345
&LabelKey=label.jst.com/type
&LabelValue=web
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<CreateNodeLabelResponse>
      <code>0</code>
      <data>
            <RequestId>869E9E27-58E8-49F1-930A-CAFBCBB5B491</RequestId>
            <Success>true</Success>
            <Code>0</Code>
            <Result>
                  <ClusterId>id1234</ClusterId>
                  <Id>0</Id>
                  <LabelKey>label.jst.com/key</LabelKey>
                  <LabelValue>value</LabelValue>
            </Result>
      </data>
      <name></name>
      <message></message>
</CreateNodeLabelResponse>
```

`JSON` 格式

```
{
    "CreateNodeLabelResponse": {
        "code": 0,
        "data": {
            "RequestId": "869E9E27-58E8-49F1-930A-CAFBCBB5B491",
            "Success": true,
            "Code": 0,
            "Result": {
                "ClusterId": "id1234",
                "Id": 0,
                "LabelKey": "label.jst.com/key",
                "LabelValue": "value"
            }
        },
        "name": "",
        "message": ""
    }
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

6.2.2. ListNodeLabels

调用ListNodeLabels查询节点标签列表。

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	ListNodeLabels	系统规定参数。取值：ListNodeLabels。
ClusterId	String	是	id1234	集群ID
PageNumber	Integer	是	1	页码，从1开始。
PageSize	Integer	是	5	页大小
LabelKey	String	否	label.jst.com/new	标签key，非必传

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回错误码
ErrorMsg	String	权限错误	错误信息
PageNumber	Integer	1	页码
PageSize	Integer	5	页大小
RequestId	String	1234	请求ID
TotalCount	Long	10	总记录数
Data	Array		返回结果
ClusterId	String	id12345	集群ID

Id	Long	1	记录ID
LabelKey	String	label.jst.com/key	标签Key
LabelValue	String	value	标签值

示例

请求示例

```
http(s)://[Endpoint]/?Action=ListNodeLabels
&ClusterId=id1234
&LabelKey=label.jst.com/new
&PageNumber=1
&PageSize=5
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<ListNodeLabelsResponse>
      <PageNumber>1</PageNumber>
      <Data>
            <LabelValue>type2</LabelValue>
            <LabelKey>label.jst.com/web</LabelKey>
            <ClusterId>c1ddd15709b01413cb6fd38f33e5a1596</ClusterId>
            <Id>2</Id>
      </Data>
      <Data>
            <LabelValue>type3</LabelValue>
            <LabelKey>label.jst.com/web1</LabelKey>
            <ClusterId>c1ddd15709b01413cb6fd38f33e5a1596</ClusterId>
            <Id>4</Id>
      </Data>
      <TotalCount>2</TotalCount>
      <PageSize>5</PageSize>
      <RequestId>2B1A4652-EB8C-4FD4-AA69-9FC8E8EEC998</RequestId>
      <Code>0</Code>
</ListNodeLabelsResponse>
```

`JSON` 格式

```
{
    "ListNodeLabelsResponse": {
        "PageNumber": 1,
        "Data": [
            {
                "LabelValue": "type2",
                "LabelKey": "label.jst.com/web",
                "ClusterId": "c1ddd15709b01413cb6fd38f33e5a1596",
                "Id": 2
            },
            {
                "LabelValue": "type3",
                "LabelKey": "label.jst.com/web1",
                "ClusterId": "c1ddd15709b01413cb6fd38f33e5a1596",
                "Id": 4
            }
        ],
        "TotalCount": 2,
        "PageSize": 5,
        "RequestId": "2B1A4652-EB8C-4FD4-AA69-9FC8E8EEC998",
        "Code": 0
    }
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

6.2.3. DeleteNodeLabel

调用DeleteNodeLabel删除节点标签。只是删除节点标签，而非解绑，删除前请保证该标签没有绑定关系。可以设置参数force=true，强制删除所有标签关联关系。

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	DeleteNodeLabel	系统规定参数。取值：DeleteNodeLabel。
ClusterId	String	是	id1234	集群ID
Force	Boolean	是	false	是否强制删除。true为强制删除，会删除关联的所有节点绑定。

LabelKey	String	是	label.jst.com/key	待删除标签Key
LabelValue	String	是	value	待删除标签value

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回错误码
ErrMsg	String	权限错误	错误信息
RequestId	String	1234	请求ID
Success	Boolean	true	是否请求成功

示例

请求示例

```
http(s)://[Endpoint]/?Action=DeleteNodeLabel
&ClusterId=id1234
&Force=false
&LabelKey=label.jst.com/key
&LabelValue=value
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<DeleteNodeLabelResponse>
      <code>3</code>
      <data>
            <ErrMsg>权限错误</ErrMsg>
            <RequestId>314BAEE2-1C6F-4AC7-BCA5-BCE9318AA8F4</RequestId>
            <Success>false</Success>
            <Code>3</Code>
      </data>
      <name>3Error</name>
      <message></message>
</DeleteNodeLabelResponse>
```

`JSON` 格式

```
{
    "DeleteNodeLabelResponse": {
        "code": 3,
        "data": {
            "ErrMsg": "权限错误",
            "RequestId": "314BAEE2-1C6F-4AC7-BCA5-BCE9318AA8F4",
            "Success": false,
            "Code": 3
        },
        "name": "3Error",
        "message": ""
    }
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

6.2.4. BindNodeLabel

调用BindNodeLabel绑定标签到Node。如果标签不存在会直接新建一个标签。

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	BindNodeLabel	系统规定参数。取值：BindNodeLabel。
ClusterId	String	是	id12345	集群实例ID
InstanceId	String	是	i-xxxxxxx	待绑定的ECS实例ID
LabelKey	String	是	label.jst.com/new	待绑定的标签Key，需以"label.jst.com/"开头。如不存在会新建
LabelValue	String	是	old	待绑定的标签value

返回数据

名称	类型	示例值	描述

Code	Integer	0	返回错误码
ErrMsg	String	权限错误	错误信息
RequestId	String	1234	请求ID
Success	Boolean	true	是否成功

示例

请求示例

```
http(s)://[Endpoint]/?Action=BindNodeLabel
&ClusterId=id12345
&InstanceId=i-xxxxxxx
&LabelKey=label.jst.com/new
&LabelValue=old
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<BindNodeLabelResponse>
      <RequestId>6704659F-E2D1-4C56-A731-3D18C3D6BD28</RequestId>
      <Success>true</Success>
      <Code>0</Code>
</BindNodeLabelResponse>
```

`JSON` 格式

```
{
    "BindNodeLabelResponse": {
        "RequestId": "6704659F-E2D1-4C56-A731-3D18C3D6BD28",
        "Success": true,
        "Code": 0
    }
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

6.2.5. UnbindNodeLabel

调用UnbindNodeLabel解绑节点标签。

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	UnbindNodeLabel	系统规定参数。取值：UnbindNodeLabel。
ClusterId	String	是	id1234	集群实例ID
InstanceId	String	是	i-yu123	待解绑ECS实例id
LabelKey	String	是	label.jst.com/key	待解绑标签Key
LabelValue	String	是	value	待解绑标签value

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回错误码
ErrMsg	String	权限错误	错误信息
RequestId	String	1234	请求id
Success	Boolean	true	是否成功

示例

请求示例

```
http(s)://[Endpoint]/?Action=UnbindNodeLabel
&ClusterId=id1234
&InstanceId=i-yu123
&LabelKey=label.jst.com/key
&LabelValue=value
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<UnbindNodeLabelResponse>
      <RequestId>6704659F-E2D1-4C56-A731-3D18C3D6BD28</RequestId>
      <Success>true</Success>
      <Code>0</Code>
</UnbindNodeLabelResponse>
```

`JSON` 格式

```
{
    "UnbindNodeLabelResponse": {
        "RequestId": "6704659F-E2D1-4C56-A731-3D18C3D6BD28",
        "Success": true,
        "Code": 0
    }
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

6.2.6. ListNodeLabelBindings

调用接口ListNodeLabelBindings查询节点标签绑定关系列表。

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	ListNodeLabelBindings	系统规定参数。取值：ListNodeLabelBindings。
ClusterId	String	是	id12345	集群实例ID，非必传
InstanceId	String	是	i-xxxxx	待查询ECS实例ID，非必传
LabelKey	String	是	label.jst.com/key	标签Key，非必传
LabelValue	String	是	value	标签Value，非必传
PageNumber	Integer	是	1	页码，从1开始
PageSize	Integer	是	5	页大小

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回错误码
ErrorMsg	String	未知错误	错误信息
PageNumber	Integer	1	页码
PageSize	Integer	5	页大小
RequestId	String	1234	请求id
TotalCount	Long	6	总记录数
Data	Array		返回结果
Id	Long	666	记录id
InstanceId	String	i-xxxxxx	ecs实例id
InstanceType	String	ecs	实例类型
LabelKey	String	label.jst.com/key	标签key
LabelValue	String	value	标签值

示例

请求示例

```
http(s)://[Endpoint]/?Action=ListNodeLabelBindings
&ClusterId=id12345
&InstanceId=i-xxxxx
&LabelKey=label.jst.com/key
&LabelValue=value
&PageNumber=1
&PageSize=5
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<PageNumber>1</PageNumber>
<Data>
    <LabelValue>node</LabelValue>
    <LabelKey>label.jst.com/type</LabelKey>
    <InstanceId>i-8vb67k4uh8713gl2aulk</InstanceId>
    <Id>9</Id>
    <InstanceType>ecs</InstanceType>
</Data>
<TotalCount>1</TotalCount>
<PageSize>5</PageSize>
<RequestId>C8E0C9EC-D6F7-4F28-8D13-B77881DDE4E3</RequestId>
<Code>0</Code>
```

`JSON` 格式

```
{
  "PageNumber": 1,
  "Data": [
    {
      "LabelValue": "node",
      "LabelKey": "label.jst.com/type",
      "InstanceId": "i-8vb67k4uh8713gl2aulk",
      "Id": 9,
      "InstanceType": "ecs"
    }
  ],
  "TotalCount": 1,
  "PageSize": 5,
  "RequestId": "C8E0C9EC-D6F7-4F28-8D13-B77881DDE4E3",
  "Code": 0
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

6.3. 集群资源分配策略

6.3.1. CreateAppResourceAlloc

调用CreateAppResourceAlloc为应用环境分配集群计算资源。

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。

请求参数

名称	类型	是否必选	示例值	描述

Action	String	是	CreateAppResourceAlloc	系统规定参数。取值：CreateAppResourceAlloc。
AppEnvId	Long	是	1234	环境id
AppId	Long	是	5555	应用id
ClusterId	String	是	id234	分配的集群实例id

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回错误码
ErrMsg	String	权限错误	错误信息
RequestId	String	12345	请求id
Success	Boolean	true	是否成功
Result	Struct		返回结果
AppEnvId	Long	1234	环境id
AppId	Long	5555	应用id
ClusterId	String	id234	集群实例id
Id	Long	1	记录id
ResourceDef	String	test	资源定义，暂不支持

示例

请求示例

```
http(s)://[Endpoint]/?AppEnvId=1234
&AppId=5555
&ClusterId=id234
&<公共请求参数>
```

正常返回示例

`XML` **格式**

```
<CreateAppResourceAlloc>
      <Result>
            <AppEnvId>2778</AppEnvId>
            <ClusterId>c1ddd15709b01413cb6fd38f33e5a1596</ClusterId>
            <AppId>8362</AppId>
            <Id>11</Id>
      </Result>
      <RequestId>88B2F939-9681-4D25-9EEB-00DD8B55F7A0</RequestId>
      <Success>true</Success>
      <Code>0</Code>
</CreateAppResourceAlloc>
```

`JSON` **格式**

```
{
    "CreateAppResourceAlloc": {
        "Result": {
            "AppEnvId": 2778,
            "ClusterId": "c1ddd15709b01413cb6fd38f33e5a1596",
            "AppId": 8362,
            "Id": 11
        },
        "RequestId": "88B2F939-9681-4D25-9EEB-00DD8B55F7A0",
        "Success": true,
        "Code": 0
    }
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

6.3.2. ListAppResourceAllocs

调用ListAppResourceAllocs查询资源绑定关系。

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	ListAppResourceAllocs	系统规定参数。取值：ListAppResourceAllocs。

PageNumber	Integer	是	1	页码，从1开始
PageSize	Integer	是	5	页大小
AppEnvId	Long	否	1234	环境id
AppId	Long	否	5555	应用id
ClusterId	String	否	id1234	集群实例id

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回错误码
ErrorMsg	String	权限失败	错误信息
PageNumber	Integer	1	页码
PageSize	Integer	5	页大小
RequestId	String	1234	请求id
TotalCount	Long	5	总记录数
Data	Array		返回结果
AppEnvId	Long	1234	环境id
AppId	Long	5555	应用id
ClusterId	String	id234	集群实例id
Id	Long	2	记录id
ResourceDef	String	test	资源定义，暂不支持

示例

请求示例

```
http(s)://[Endpoint]/?Action=ListAppResourceAllocs
&AppEnvId=1234
&AppId=5555
&ClusterId=id1234
&PageNumber=1
&PageSize=5
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<ListAppResourceAllocsResponse>
      <PageNumber>1</PageNumber>
      <Data>
            <AppEnvId>2344</AppEnvId>
            <ClusterId>c64221d415e234a7982eaf00a8162855c</ClusterId>
            <AppId>7725</AppId>
            <Id>6</Id>
      </Data>
      <Data>
            <AppEnvId>2698</AppEnvId>
            <ClusterId>c64221d415e234a7982eaf00a8162855c</ClusterId>
            <AppId>8247</AppId>
            <Id>7</Id>
      </Data>
      <TotalCount>5</TotalCount>
      <PageSize>2</PageSize>
      <RequestId>52608F2A-FA83-4F71-9A44-0ED4B7A35B3D</RequestId>
      <Code>0</Code>
</ListAppResourceAllocsResponse>
```

`JSON` 格式

```
{
    "ListAppResourceAllocsResponse": {
        "PageNumber": 1,
        "Data": [
            {
                "AppEnvId": 2344,
                "ClusterId": "c64221d415e234a7982eaf00a8162855c",
                "AppId": 7725,
                "Id": 6
            },
            {
                "AppEnvId": 2698,
                "ClusterId": "c64221d415e234a7982eaf00a8162855c",
                "AppId": 8247,
                "Id": 7
            }
        ],
        "TotalCount": 5,
        "PageSize": 2,
        "RequestId": "52608F2A-FA83-4F71-9A44-0ED4B7A35B3D",
        "Code": 0
    }
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

6.3.3. DeleteAppResourceAlloc

调用DeleteAppResourceAlloc删除应用环境与集群绑定关系。

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	DeleteAppResourceAlloc	系统规定参数。取值：DeleteAppResourceAlloc。
AppEnvId	Long	是	1234	环境id

返回数据

名称	类型	示例值	描述

Code	Integer	0	返回错误码
ErrMsg	String	权限错误	错误信息
RequestId	String	1234	请求id
Success	Boolean	true	是否成功

示例

请求示例

```
http(s)://[Endpoint]/?Action=DeleteAppResourceAlloc
&AppEnvId=1234
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<DeleteAppResourceAllocResponse>
      <RequestId>37D67DA3-7564-403B-A200-F08E8386EAB1</RequestId>
      <Success>true</Success>
      <Code>0</Code>
</DeleteAppResourceAllocResponse>
```

`JSON` 格式

```
{
    "DeleteAppResourceAllocResponse": {
        "RequestId": "37D67DA3-7564-403B-A200-F08E8386EAB1",
        "Success": true,
        "Code": 0
    }
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

6.3.4. DescribeAppResourceAlloc

调用DescribeAppResourceAlloc查询环境绑定的集群资源

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	DescribeAppResourceAlloc	系统规定参数。取值：DescribeAppResourceAlloc。
AppEnvId	Long	是	1234	环境id

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回错误码
ErrMsg	String	权限错误	错误信息
RequestId	String	1234	请求id
Success	Boolean	true	是否成功
Result	Struct		返回结果
AppEnvId	Long	1234	环境id
AppId	Long	5555	应用id
ClusterId	String	id1234	容器集群实例id
Id	Long	0	记录id
ResourceDef	String	test	资源定义，暂不支持

示例

请求示例

```
http(s)://[Endpoint]/?Action=DescribeAppResourceAlloc
&AppEnvId=1234
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<DescribeAppResourceAllocResponse>
      <Result>
            <AppEnvId>2778</AppEnvId>
            <ClusterId>c1ddd15709b01413cb6fd38f33e5a1596</ClusterId>
            <AppId>8362</AppId>
            <Id>11</Id>
      </Result>
      <RequestId>37D67DA3-7564-403B-A200-F08E8386EAB1</RequestId>
      <Success>true</Success>
      <Code>0</Code>
</DescribeAppResourceAllocResponse>
```

`JSON` 格式

```
{
    "DescribeAppResourceAllocResponse": {
        "Result": {
            "AppEnvId": 2778,
            "ClusterId": "c1ddd15709b01413cb6fd38f33e5a1596",
            "AppId": 8362,
            "Id": 11
        },
        "RequestId": "37D67DA3-7564-403B-A200-F08E8386EAB1",
        "Success": true,
        "Code": 0
    }
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

6.4. 集群管理

6.4.1. ListCluster

调用ListCluster接口查询集群列表

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	ListCluster	系统规定参数。取值：ListCluster。

BusinessCode	String	否	JST	业务域 JST | NEW_RETAIL | MINI_APP | SUPPLY | MESSAGE
EnvType	String	否	PRO	环境类型 TEST|PRO
PageNum	Integer	否	1	分页页码
PageSize	Integer	否	20	分页大小

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码
ErrorMsg	String	error	错误信息
PageNumber	Integer	1	分页页码
PageSize	Integer	20	分页大小
RequestId	String	hjhlkjh	请求id
TotalCount	Long	1	总数
Data	Array		列表数据
BusinessCode	String	JST	业务域
ClusterTitle	String	test	集群名称
CreateStatus	String	1	创建状态
EnvType	String	1	环境类型
Id	Long	1	主键id
InstanceId	String	jgjhjlhjkl	实例id

KeyPair	String	test	keypair
NetPlug	String	terway	网络插件
Password	String	*********	密码
PodCIDR	String	172.29.0.0/16	pod 网段
RegionId	String	cn-zhangjiakou	地域
RegionName	String	cn-zhangjiakou	地域
ServiceCIDR	String	172.29.0.0/16	service网段
Status	String	running	集群状态
VpcId	String	vpc-hjhjhj	vpc
WorkLoadCpu	String	0.78	集群cpu水位(已请求)
WorkLoadMem	String	0.79	集群内存水位(已请求)
EcsIds	List	["i-jhjhj"]	ECS实例id列表
VswitchIds	List	["v-hhghgh"]	交换机

示例

请求示例

```
http(s)://[Endpoint]/?Action=ListCluster
&BusinessCode=JST
&EnvType=PRO
&PageNum=1
&PageSize=20
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<ListClusterResponse>
      <TotalCount>1</TotalCount>
      <PageSize>20</PageSize>
      <RequestId>3378D780-16B5-4661-B69B-5678CF0ACD11</RequestId>
      <PageNumber>1</PageNumber>
      <Data>
            <Status>running</Status>
            <ClusterTitle>测试环境集群</ClusterTitle>
            <WorkLoadCpu>0.14</WorkLoadCpu>
            <KeyPair></KeyPair>
            <NetPlug>Terway</NetPlug>
            <PodCIDR>172.22.0.0/16</PodCIDR>
            <InstanceId>c5615b8c95b2341f8937ff11693ddf3bd</InstanceId>
            <EcsIds>i-8vb1nz5l71xubb6rjnle</EcsIds>
            <EcsIds>i-8vb1nz5l71xubb6rjnld</EcsIds>
            <EcsIds>i-8vb1nz5l71xubb6rjnlf</EcsIds>
            <EcsIds>i-8vb1nz5l71xubb6rjnlc</EcsIds>
            <BusinessCode>JST</BusinessCode>
            <EnvType>0</EnvType>
            <WorkLoadMem>0.07</WorkLoadMem>
            <VpcId>vpc-8vbm6coyrh92bbjj1ej0n</VpcId>
            <RegionName>cn-zhangjiakou</RegionName>
            <VswitchIds>vsw-8vbq2k29mzi65lo7c6lzr</VswitchIds>
            <VswitchIds>vsw-8vbl8s66ry10ch52hw3u4</VswitchIds>
            <ServiceCIDR>172.23.0.0/20</ServiceCIDR>
            <Id>138</Id>
            <RegionId>cn-zhangjiakou</RegionId>
            <Password>*********</Password>
      </Data>
      <Code>0</Code>
</ListClusterResponse>
```


`JSON` **格式**

```
{
    "TotalCount": 1,
    "PageSize": 20,
    "RequestId": "3378D780-16B5-4661-B69B-5678CF0ACD11",
    "PageNumber": 1,
    "Data": [
      {
        "Status": "running",
        "ClusterTitle": "测试环境集群",
        "WorkLoadCpu": 0.14,
        "KeyPair": "",
        "NetPlug": "Terway",
        "PodCIDR": "172.22.0.0/16",
        "InstanceId": "c5615b8c95b2341f8937ff11693ddf3bd",
        "EcsIds": [
          "i-8vb1nz5l71xubb6rjnle",
          "i-8vb1nz5l71xubb6rjnld",
          "i-8vb1nz5l71xubb6rjnlf",
          "i-8vb1nz5l71xubb6rjnlc"
        ],
        "BusinessCode": "JST",
        "EnvType": 0,
        "WorkLoadMem": 0.07,
        "VpcId": "vpc-8vbm6coyrh92bbjj1ej0n",
        "RegionName": "cn-zhangjiakou",
        "VswitchIds": [
          "vsw-8vbq2k29mzi65lo7c6lzr",
          "vsw-8vbl8s66ry10ch52hw3u4"
        ],
        "ServiceCIDR": "172.23.0.0/20",
        "Id": 138,
        "RegionId": "cn-zhangjiakou",
        "Password": "Taeroot1234!"
      }
    ],
    "Code": 0
  }
```


错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

6.4.2. DeleteCluster

调用DeleteCluster接口删除集群

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	DeleteCluster	系统规定参数。取值：DeleteCluster。
ClusterInstanceId	String	是	ajkj4009fff	集群实例id

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码 0表示成功，其他表示失败
ErrMsg	String	删除集群失败	错误信息
RequestId	String	QJSD-4HJHJK-KHJH	requesId
Success	Boolean	true	是否成功
Result	Struct		结果体
Nonsense	Integer	1	暂无意义

示例

请求示例

```
http(s)://[Endpoint]/?Action=DeleteCluster
&ClusterInstanceId=ajkj4009fff
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<DeleteClusterResponse>
      <Result>
            <Nonsense>1</Nonsense>
      </Result>
      <RequestId>8BB49E3E-075C-4D4E-B6CC-7F7FBCFDE5D4</RequestId>
      <Success>true</Success>
      <Code>0</Code>
</DeleteClusterResponse>
```

`JSON` 格式

```
{
    "Result": {
      "Nonsense": 1
    },
    "RequestId": "8BB49E3E-075C-4D4E-B6CC-7F7FBCFDE5D4",
    "Success": true,
    "Code": 0
  }
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

6.4.3. ListClusterNode

调用ListClusterNode接口查询集群中节点列表

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	ListClusterNode	系统规定参数。取值：ListClusterNode。
ClusterInstanceId	String	是	ddd	集群实例id
PageNum	Integer	否	1	分页页码
PageSize	Integer	否	20	分页页码

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码
ErrorMsg	String	error	错误信息
PageNumber	Integer	1	分页页码
PageSize	Integer	20	分页大小

RequestId	String	4GHV-HHHG	请求id
TotalCount	Long	7	总数
Data	Array		列表数据
OSName	String	CentOS 7.0	操作系统版本 如 CentOS 7.0 64位
BusinessCode	String	JST	业务域
EcsConfiguration	String	4vcpu 8G	资源配置 如4核8G(I/O优化)
EcsCpu	String	8	核数
EcsEip	String	49.198.33.10	弹性公网ip
EcsExpiredTime	String	2099-12-31	过期时间
EcsLocalStorageCapacity	String	40	本地存储容量
EcsMemory	String	8192	内存
EcsOsType	String	linux	操作系统类型
EcsPrivateIp	String	192.168.48.6	私网ip
EcsPublicIp	String	192.168.48.6	公网ip
EcsZone	String	cn-zhangjiakou	可用区
InstanceId	String	i-hgjkghjg	实例id
InstanceName	String	worker-k8s-for-cs-c6fd41efd9df74f50a8e7804fcb73c27a	实例名称
InstanceNetworkType	String	vpc	网络类型
InstanceType	String	ecs.c6.xlarge	实例规格

InternetMaxBandwidthIn	String	3	公网入带宽最大值(单位：Mbps)
InternetMaxBandwidthOut	String	3	公网出带宽最大值(单位：Mbps)
RegionId	String	cn-zhangjiakou	地域
VpcId	String	vpc-hjhjhj	vpc

示例

请求示例

```
http(s)://[Endpoint]/?Action=ListClusterNode
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<ListClusterNodeResponse>
      <TotalCount>7</TotalCount>
      <PageSize>20</PageSize>
      <RequestId>84A93553-B470-4B5E-BA9C-9A4C97CF4AA3</RequestId>
      <PageNumber>1</PageNumber>
      <Data>
            <EcsPrivateIp>192.168.48.7</EcsPrivateIp>
            <InstanceId>i-8vb4n7it53012zup5gjr</InstanceId>
            <EcsMemory>8192</EcsMemory>
            <EcsOsType>linux</EcsOsType>
            <InstanceNetworkType>vpc</InstanceNetworkType>
            <InstanceName>worker-k8s-for-cs-
c6fd41efd9df74f50a8e7804fcb73c27a</InstanceName>
            <InternetMaxBandwidthOut>1</InternetMaxBandwidthOut>
            <InternetMaxBandwidthIn>1500</InternetMaxBandwidthIn>
            <VpcId>vpc-8vbm6coyrh92bbjj1ej0n</VpcId>
            <EcsCpu>4</EcsCpu>
            <EcsPublicIp>39.99.251.27</EcsPublicIp>
            <EcsExpiredTime>2099-12-31</EcsExpiredTime>
            <RegionId>cn-zhangjiakou</RegionId>
            <InstanceType>ecs.c6.xlarge</InstanceType>
            <EcsZone>cn-zhangjiakou</EcsZone>
            <EcsConfiguration>4vcpu 8G</EcsConfiguration>
      </Data>
      <Data>
            <EcsPrivateIp>192.168.48.6</EcsPrivateIp>
            <InstanceId>i-8vb4n7it53012zup5gjs</InstanceId>
            <EcsMemory>8192</EcsMemory>
            <EcsOsType>linux</EcsOsType>
            <InstanceNetworkType>vpc</InstanceNetworkType>
            <InstanceName>worker-k8s-for-cs-
c6fd41efd9df74f50a8e7804fcb73c27a</InstanceName>
```

```
            <InternetMaxBandwidthOut>1</InternetMaxBandwidthOut>
            <InternetMaxBandwidthIn>1500</InternetMaxBandwidthIn>
            <VpcId>vpc-8vbm6coyrh92bbjj1ej0n</VpcId>
            <EcsCpu>4</EcsCpu>
            <EcsPublicIp>39.98.120.55</EcsPublicIp>
            <EcsExpiredTime>2099-12-31</EcsExpiredTime>
            <RegionId>cn-zhangjiakou</RegionId>
            <InstanceType>ecs.c6.xlarge</InstanceType>
            <EcsZone>cn-zhangjiakou</EcsZone>
            <EcsConfiguration>4vcpu 8G</EcsConfiguration>
      </Data>
      <Data>
            <EcsPrivateIp>192.168.49.234</EcsPrivateIp>
            <InstanceId>i-8vbdorsyr108yfuskhtz</InstanceId>
            <EcsMemory>8192</EcsMemory>
            <EcsOsType>linux</EcsOsType>
            <InstanceNetworkType>vpc</InstanceNetworkType>
            <InstanceName>worker-k8s-for-cs-
c6fd41efd9df74f50a8e7804fcb73c27a</InstanceName>
            <InternetMaxBandwidthOut>5</InternetMaxBandwidthOut>
            <InternetMaxBandwidthIn>1500</InternetMaxBandwidthIn>
            <VpcId>vpc-8vbm6coyrh92bbjj1ej0n</VpcId>
            <EcsCpu>4</EcsCpu>
            <EcsPublicIp>39.98.254.41</EcsPublicIp>
            <EcsExpiredTime>2099-12-31</EcsExpiredTime>
            <RegionId>cn-zhangjiakou</RegionId>
            <InstanceType>ecs.c6.xlarge</InstanceType>
            <EcsZone>cn-zhangjiakou</EcsZone>
            <EcsConfiguration>4vcpu 8G</EcsConfiguration>
      </Data>
      <Data>
            <EcsPrivateIp>192.168.49.232</EcsPrivateIp>
            <InstanceId>i-8vbdorsyr108yfuskhu0</InstanceId>
            <EcsMemory>8192</EcsMemory>
            <EcsOsType>linux</EcsOsType>
            <InstanceNetworkType>vpc</InstanceNetworkType>
            <InstanceName>worker-k8s-for-cs-
c6fd41efd9df74f50a8e7804fcb73c27a</InstanceName>
            <InternetMaxBandwidthOut>5</InternetMaxBandwidthOut>
            <InternetMaxBandwidthIn>1500</InternetMaxBandwidthIn>
            <VpcId>vpc-8vbm6coyrh92bbjj1ej0n</VpcId>
            <EcsCpu>4</EcsCpu>
            <EcsPublicIp>39.100.54.126</EcsPublicIp>
            <EcsExpiredTime>2099-12-31</EcsExpiredTime>
            <RegionId>cn-zhangjiakou</RegionId>
            <InstanceType>ecs.c6.xlarge</InstanceType>
            <EcsZone>cn-zhangjiakou</EcsZone>
            <EcsConfiguration>4vcpu 8G</EcsConfiguration>
      </Data>
      <Data>
            <EcsPrivateIp>192.168.49.233</EcsPrivateIp>
            <InstanceId>i-8vbdorsyr108yfuskhu1</InstanceId>
            <EcsMemory>8192</EcsMemory>
            <EcsOsType>linux</EcsOsType>
```

```
            <InstanceNetworkType>vpc</InstanceNetworkType>
            <InstanceName>worker-k8s-for-cs-
c6fd41efd9df74f50a8e7804fcb73c27a</InstanceName>
            <InternetMaxBandwidthOut>5</InternetMaxBandwidthOut>
            <InternetMaxBandwidthIn>1500</InternetMaxBandwidthIn>
            <VpcId>vpc-8vbm6coyrh92bbjj1ej0n</VpcId>
            <EcsCpu>4</EcsCpu>
            <EcsPublicIp>39.100.39.33</EcsPublicIp>
            <EcsExpiredTime>2099-12-31</EcsExpiredTime>
            <RegionId>cn-zhangjiakou</RegionId>
            <InstanceType>ecs.c6.xlarge</InstanceType>
            <EcsZone>cn-zhangjiakou</EcsZone>
            <EcsConfiguration>4vcpu 8G</EcsConfiguration>
      </Data>
      <Data>
            <EcsPrivateIp>192.168.49.235</EcsPrivateIp>
            <InstanceId>i-8vbdorsyr108yfuskhu2</InstanceId>
            <EcsMemory>8192</EcsMemory>
            <EcsOsType>linux</EcsOsType>
            <InstanceNetworkType>vpc</InstanceNetworkType>
            <InstanceName>worker-k8s-for-cs-
c6fd41efd9df74f50a8e7804fcb73c27a</InstanceName>
            <InternetMaxBandwidthOut>5</InternetMaxBandwidthOut>
            <InternetMaxBandwidthIn>1500</InternetMaxBandwidthIn>
            <VpcId>vpc-8vbm6coyrh92bbjj1ej0n</VpcId>
            <EcsCpu>4</EcsCpu>
            <EcsPublicIp>39.100.2.236</EcsPublicIp>
            <EcsExpiredTime>2099-12-31</EcsExpiredTime>
            <RegionId>cn-zhangjiakou</RegionId>
            <InstanceType>ecs.c6.xlarge</InstanceType>
            <EcsZone>cn-zhangjiakou</EcsZone>
            <EcsConfiguration>4vcpu 8G</EcsConfiguration>
      </Data>
      <Data>
            <EcsPrivateIp>192.168.49.243</EcsPrivateIp>
            <InstanceId>i-8vbimgblfznzjftuiks8</InstanceId>
            <EcsMemory>16384</EcsMemory>
            <EcsOsType>linux</EcsOsType>
            <InstanceNetworkType>vpc</InstanceNetworkType>
            <InstanceName>launch-advisor-20191107</InstanceName>
            <InternetMaxBandwidthOut>1</InternetMaxBandwidthOut>
            <InternetMaxBandwidthIn>2500</InternetMaxBandwidthIn>
            <VpcId>vpc-8vbm6coyrh92bbjj1ej0n</VpcId>
            <EcsCpu>8</EcsCpu>
            <EcsPublicIp>39.99.134.192</EcsPublicIp>
            <EcsExpiredTime>2099-12-31</EcsExpiredTime>
            <RegionId>cn-zhangjiakou</RegionId>
            <InstanceType>ecs.c6.2xlarge</InstanceType>
            <EcsZone>cn-zhangjiakou</EcsZone>
            <EcsConfiguration>8vcpu 16G</EcsConfiguration>
      </Data>
      <Code>0</Code>
</ListClusterNodeResponse>
```

JSON 格式


```
{
    "TotalCount": 7,
    "PageSize": 20,
    "RequestId": "84A93553-B470-4B5E-BA9C-9A4C97CF4AA3",
    "PageNumber": 1,
    "Data": [
      {
        "EcsPrivateIp": "192.168.48.7",
        "InstanceId": "i-8vb4n7it53012zup5gjr",
        "EcsMemory": "8192",
        "EcsOsType": "linux",
        "InstanceNetworkType": "vpc",
        "InstanceName": "worker-k8s-for-cs-c6fd41efd9df74f50a8e7804fcb73c27a",
        "InternetMaxBandwidthOut": "1",
        "InternetMaxBandwidthIn": "1500",
        "VpcId": "vpc-8vbm6coyrh92bbjj1ej0n",
        "EcsCpu": "4",
        "EcsPublicIp": "39.99.251.27",
        "EcsExpiredTime": "2099-12-31",
        "RegionId": "cn-zhangjiakou",
        "InstanceType": "ecs.c6.xlarge",
        "EcsZone": "cn-zhangjiakou",
        "EcsConfiguration": "4vcpu 8G"
      },
      {
        "EcsPrivateIp": "192.168.48.6",
        "InstanceId": "i-8vb4n7it53012zup5gjs",
        "EcsMemory": "8192",
        "EcsOsType": "linux",
        "InstanceNetworkType": "vpc",
        "InstanceName": "worker-k8s-for-cs-c6fd41efd9df74f50a8e7804fcb73c27a",
        "InternetMaxBandwidthOut": "1",
        "InternetMaxBandwidthIn": "1500",
        "VpcId": "vpc-8vbm6coyrh92bbjj1ej0n",
        "EcsCpu": "4",
        "EcsPublicIp": "39.98.120.55",
        "EcsExpiredTime": "2099-12-31",
        "RegionId": "cn-zhangjiakou",
        "InstanceType": "ecs.c6.xlarge",
        "EcsZone": "cn-zhangjiakou",
        "EcsConfiguration": "4vcpu 8G"
      },
      {
        "EcsPrivateIp": "192.168.49.234",
        "InstanceId": "i-8vbdorsyr108yfuskhtz",
        "EcsMemory": "8192",
        "EcsOsType": "linux",
        "InstanceNetworkType": "vpc",
        "InstanceName": "worker-k8s-for-cs-c6fd41efd9df74f50a8e7804fcb73c27a",
        "InternetMaxBandwidthOut": "5",
        "InternetMaxBandwidthIn": "1500",
        "VpcId": "vpc-8vbm6coyrh92bbjj1ej0n",
```

```
        "EcsCpu": "4",
        "EcsPublicIp": "39.98.254.41",
        "EcsExpiredTime": "2099-12-31",
        "RegionId": "cn-zhangjiakou",
        "InstanceType": "ecs.c6.xlarge",
        "EcsZone": "cn-zhangjiakou",
        "EcsConfiguration": "4vcpu 8G"
      },
      {
        "EcsPrivateIp": "192.168.49.232",
        "InstanceId": "i-8vbdorsyr108yfuskhu0",
        "EcsMemory": "8192",
        "EcsOsType": "linux",
        "InstanceNetworkType": "vpc",
        "InstanceName": "worker-k8s-for-cs-c6fd41efd9df74f50a8e7804fcb73c27a",
        "InternetMaxBandwidthOut": "5",
        "InternetMaxBandwidthIn": "1500",
        "VpcId": "vpc-8vbm6coyrh92bbjj1ej0n",
        "EcsCpu": "4",
        "EcsPublicIp": "39.100.54.126",
        "EcsExpiredTime": "2099-12-31",
        "RegionId": "cn-zhangjiakou",
        "InstanceType": "ecs.c6.xlarge",
        "EcsZone": "cn-zhangjiakou",
        "EcsConfiguration": "4vcpu 8G"
      },
      {
        "EcsPrivateIp": "192.168.49.233",
        "InstanceId": "i-8vbdorsyr108yfuskhu1",
        "EcsMemory": "8192",
        "EcsOsType": "linux",
        "InstanceNetworkType": "vpc",
        "InstanceName": "worker-k8s-for-cs-c6fd41efd9df74f50a8e7804fcb73c27a",
        "InternetMaxBandwidthOut": "5",
        "InternetMaxBandwidthIn": "1500",
        "VpcId": "vpc-8vbm6coyrh92bbjj1ej0n",
        "EcsCpu": "4",
        "EcsPublicIp": "39.100.39.33",
        "EcsExpiredTime": "2099-12-31",
        "RegionId": "cn-zhangjiakou",
        "InstanceType": "ecs.c6.xlarge",
        "EcsZone": "cn-zhangjiakou",
        "EcsConfiguration": "4vcpu 8G"
      },
      {
        "EcsPrivateIp": "192.168.49.235",
        "InstanceId": "i-8vbdorsyr108yfuskhu2",
        "EcsMemory": "8192",
        "EcsOsType": "linux",
        "InstanceNetworkType": "vpc",
        "InstanceName": "worker-k8s-for-cs-c6fd41efd9df74f50a8e7804fcb73c27a",
        "InternetMaxBandwidthOut": "5",
        "InternetMaxBandwidthIn": "1500",
        "VpcId": "vpc-8vbm6coyrh92bbjj1ej0n",
        "EcsCpu": "4",
```

```
        "EcsCpu": "4",
        "EcsPublicIp": "39.100.2.236",
        "EcsExpiredTime": "2099-12-31",
        "RegionId": "cn-zhangjiakou",
        "InstanceType": "ecs.c6.xlarge",
        "EcsZone": "cn-zhangjiakou",
        "EcsConfiguration": "4vcpu 8G"
      },
      {
        "EcsPrivateIp": "192.168.49.243",
        "InstanceId": "i-8vbimgblfznzjftuiks8",
        "EcsMemory": "16384",
        "EcsOsType": "linux",
        "InstanceNetworkType": "vpc",
        "InstanceName": "launch-advisor-20191107",
        "InternetMaxBandwidthOut": "1",
        "InternetMaxBandwidthIn": "2500",
        "VpcId": "vpc-8vbm6coyrh92bbjj1ej0n",
        "EcsCpu": "8",
        "EcsPublicIp": "39.99.134.192",
        "EcsExpiredTime": "2099-12-31",
        "RegionId": "cn-zhangjiakou",
        "InstanceType": "ecs.c6.2xlarge",
        "EcsZone": "cn-zhangjiakou",
        "EcsConfiguration": "8vcpu 16G"
      }
    ],
    "Code": 0
  }
```


错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

6.4.4. QueryClusterDetail

调用QueryClusterDetail接口查询集群详情

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	QueryClusterDetail	系统规定参数。取值：QueryClusterDetail。
ClusterInstanceId	String	是	jhjhj	集群实例id

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码
ErrMsg	String	error	错误信息
RequestId	String	djkjk	请求id
Success	Boolean	true	是否成功
Result	Struct		数据详情
NodeWorkLoadList	List	[]	每个节点工作负载
BasicInfo	Struct		集群基本信息
BusinessCode	String	JST	业务域
ClusterId	Long	12	主键id
ClusterInstanceId	String	jhkhjk	集群实例id
ClusterName	String	test	集群名称
EnvType	String	1	环境类型
HasInstallArmsPilot	Boolean	false	是否安装arms组件
HasInstallLogController	Boolean	false	是否安装sls组件
InstallArmsInProcess	Boolean	false	是否安装中
InstallLogInProcess	Boolean	false	是否安装中
MainUserId	String	989898	用户id
RegionId	String	cn-zhangjiakou	地域

RegionName	String	cn-zhangjiakou	地域
UserNick	String	test	nick
VpcId	String	vpc-test	vpc
EcsIds	List	["i-hhjhg"]	集群主机id
Vswitchs	List	["v-ghgh"]	交换机
InstanceInfo	Struct		集群中容器实例信息
AllocatePodCount	Integer	1	已部署POD数量
AppCount	Integer	1	部署应用总数
AllocatedPodInfoList	List	[]	集群中已部署pod信息
AvailablePodInfoList	List	[]	还可以部署的容器规格以及数量 仅供参考
NetInfo	Struct		集群网络信息
NetPlugType	String	terway	网络插件类型
ProdCIDR	String	172.30.0.0/20	pod网段
ServiceCIDR	String	172.30.0.0/20	service网段
WorkLoad	Struct		集群负载
AllNodeCount	Integer	4	节点总数
AllocateAllPodCount	Integer	20	集群内所有POD总数
AllocateAppPodCount	Integer	10	应用自身的POD总数
CpuCapacityTotal	String	4	总核数

CpuLevel	String	0.75	cpu水位
CpuRequest	String	3	cpu请求核数
CpuRequestPercent	String	0.75	cpu请求比例
CpuTotal	String	4	cpu总核数
CpuUse	String	1	cpu已使用核数
CpuUsePercent	String	0.25	cpu使用率
MemCapacityTotal	String	40000	内存总数 字节数
MemLevel	String	0.75	集群内存水位
MemRequest	String	30000	内存请求总数 字节数
MemRequestPercent	String	0.75	内存已请求比例
MemTotal	String	40000	内存总数 字节数
MemUse	String	10000	内存使用总数 字节数
MemUsePercent	String	0.25	内存使用率

示例

请求示例

```
http(s)://[Endpoint]/?Action=QueryClusterDetail
&ClusterInstanceId=jhjhj
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<QueryClusterDetailResponse>
      <RequestId>16A78EEB-E502-4DC5-A9A0-9969980F1B58</RequestId>
      <Code>0</Code>
      <Success>true</Success>
      <Result>
            <BasicInfo>
```

```
<ClusterInstanceId>c6fd41efd9df74f50a8e7804fcb73c27a</ClusterInstanceId>
                  <InstallLogInProcess>false</InstallLogInProcess>
                  <HasInstallLogController>true</HasInstallLogController>
                  <Vswitchs>vsw-8vbr9ukajnfj1i98otgpn</Vswitchs>
                  <Vswitchs>vsw-8vbq2k29mzi65lo7c6lzr</Vswitchs>
                  <Vswitchs>vsw-8vbl8s66ry10ch52hw3u4</Vswitchs>
                  <HasInstallArmsPilot>true</HasInstallArmsPilot>
                  <VpcId>vpc-8vbm6coyrh92bbjj1ej0n</VpcId>
                  <RegionName>cn-zhangjiakou</RegionName>
                  <InstallArmsInProcess>false</InstallArmsInProcess>
                  <ClusterName>杭羽测试-正式环境</ClusterName>
                  <RegionId>cn-zhangjiakou</RegionId>
            </BasicInfo>
            <WorkLoad>
                  <MemLevel>0</MemLevel>
                  <AllocateAppPodCount>17</AllocateAppPodCount>
                  <CpuRequestPercent>0.57</CpuRequestPercent>
                  <MemRequest>25769803776</MemRequest>
                  <MemUsePercent>0.46</MemUsePercent>
                  <CpuUsePercent>0.02</CpuUsePercent>
                  <CpuUse>0.69</CpuUse>
                  <MemTotal>56292110336</MemTotal>
                  <CpuLevel>0</CpuLevel>
                  <MemUse>25901572096</MemUse>
                  <MemRequestPercent>0.46</MemRequestPercent>
                  <AllNodeCount>7</AllNodeCount>
                  <AllocateAllPodCount>72</AllocateAllPodCount>
                  <CpuCapacityTotal>32</CpuCapacityTotal>
                  <MemCapacityTotal>63632142336</MemCapacityTotal>
                  <CpuRequest>18.12</CpuRequest>
                  <CpuTotal>32</CpuTotal>
            </WorkLoad>
            <InstanceInfo>
                  <AllocatedPodInfoList>
                        <appName>image_php44</appName>
                        <envName>正式环境</envName>
                        <memRequest>1GB</memRequest>
                        <appId>5687</appId>
                        <cupRequest>1vcpu</cupRequest>
                        <podCount>1</podCount>
                        <envId>1136</envId>
                  </AllocatedPodInfoList>
                  <AllocatedPodInfoList>
                        <appName>image_nodejs</appName>
                        <envName>正式环境</envName>
                        <memRequest>512MB</memRequest>
                        <appId>5691</appId>
                        <cupRequest>0.5vcpu</cupRequest>
                        <podCount>1</podCount>
                        <envId>1139</envId>
                  </AllocatedPodInfoList>
                  <AllocatedPodInfoList>
                        <appName>image_php55</appName>
```

```
            <envName>正式环境</envName>
            <memRequest>1GB</memRequest>
            <appId>5685</appId>
            <cupRequest>1vcpu</cupRequest>
            <podCount>1</podCount>
            <envId>1132</envId>
        </AllocatedPodInfoList>
        <AllocatedPodInfoList>
            <appName>imgae_php72_newnew</appName>
            <envName>正式环境</envName>
            <memRequest>512MB</memRequest>
            <appId>6063</appId>
            <cupRequest>0.5vcpu</cupRequest>
            <podCount>1</podCount>
            <envId>1420</envId>
        </AllocatedPodInfoList>
        <AllocatedPodInfoList>
            <appName>springboot2_prom</appName>
            <envName>正式环境</envName>
            <memRequest>1500MB</memRequest>
            <appId>6974</appId>
            <cupRequest>1vcpu</cupRequest>
            <podCount>2</podCount>
            <envId>1874</envId>
        </AllocatedPodInfoList>
        <AllocatedPodInfoList>
            <appName>image_php72_new</appName>
            <envName>正式环境</envName>
            <memRequest>256MB</memRequest>
            <appId>6017</appId>
            <cupRequest>0.5vcpu</cupRequest>
            <podCount>1</podCount>
            <envId>1385</envId>
        </AllocatedPodInfoList>
        <AllocatedPodInfoList>
            <appName>image_springboot</appName>
            <envName>正式环境</envName>
            <memRequest>2GB</memRequest>
            <appId>5665</appId>
            <cupRequest>1vcpu</cupRequest>
            <podCount>1</podCount>
            <envId>1116</envId>
        </AllocatedPodInfoList>
        <AllocatedPodInfoList>
            <appName>imae_springboot2</appName>
            <envName>正式环境</envName>
            <memRequest>1GB</memRequest>
            <appId>5819</appId>
            <cupRequest>1vcpu</cupRequest>
            <podCount>1</podCount>
            <envId>1270</envId>
        </AllocatedPodInfoList>
        <AllocatedPodInfoList>
            <appName>cms_test</appName>
            <envName>正式环境</envName>
```

```
            <envName>正式环境</envName>
            <memRequest>1GB</memRequest>
            <appId>5926</appId>
            <cupRequest>0.5vcpu</cupRequest>
            <podCount>1</podCount>
            <envId>1323</envId>
        </AllocatedPodInfoList>
        <AllocatedPodInfoList>
            <appName>java_prom</appName>
            <envName>正式环境</envName>
            <memRequest>1500MB</memRequest>
            <appId>6945</appId>
            <cupRequest>1vcpu</cupRequest>
            <podCount>2</podCount>
            <envId>1863</envId>
        </AllocatedPodInfoList>
        <AllocatedPodInfoList>
            <appName>image_php72</appName>
            <envName>正式环境</envName>
            <memRequest>512MB</memRequest>
            <appId>6015</appId>
            <cupRequest>0.5vcpu</cupRequest>
            <podCount>1</podCount>
            <envId>1383</envId>
        </AllocatedPodInfoList>
        <AllocatedPodInfoList>
            <appName>cms_group</appName>
            <envName>正式环境</envName>
            <memRequest>512MB</memRequest>
            <appId>5927</appId>
            <cupRequest>0.5vcpu</cupRequest>
            <podCount>1</podCount>
            <envId>1325</envId>
        </AllocatedPodInfoList>
        <AllocatedPodInfoList>
            <appName>add</appName>
            <envName>正式环境</envName>
            <memRequest>4900MB</memRequest>
            <appId>5821</appId>
            <cupRequest>1.5vcpu</cupRequest>
            <podCount>1</podCount>
            <envId>1273</envId>
        </AllocatedPodInfoList>
        <AllocatedPodInfoList>
            <appName>image_nodejs4</appName>
            <envName>正式环境</envName>
            <memRequest>500MB</memRequest>
            <appId>5702</appId>
            <cupRequest>0.5vcpu</cupRequest>
            <podCount>1</podCount>
            <envId>1152</envId>
        </AllocatedPodInfoList>
        <AllocatedPodInfoList>
            <appName>image_php56</appName>
            <envName>正式环境</envName>
```

```
                        <memRequest>1GB</memRequest>
                        <appId>5674</appId>
                        <cupRequest>1vcpu</cupRequest>
                        <podCount>1</podCount>
                        <envId>1124</envId>
                  </AllocatedPodInfoList>
                  <AllocatePodCount>17</AllocatePodCount>
                  <AppCount>15</AppCount>
                  <AvailablePodInfoList>
                        <memRequest>1</memRequest>
                        <cupRequest>1</cupRequest>
                        <availablePodCount>9</availablePodCount>
                  </AvailablePodInfoList>
                  <AvailablePodInfoList>
                        <memRequest>4</memRequest>
                        <cupRequest>2</cupRequest>
                        <availablePodCount>4</availablePodCount>
                  </AvailablePodInfoList>
                  <AvailablePodInfoList>
                        <memRequest>8</memRequest>
                        <cupRequest>4</cupRequest>
                        <availablePodCount>1</availablePodCount>
                  </AvailablePodInfoList>
                  <AvailablePodInfoList>
                        <memRequest>16</memRequest>
                        <cupRequest>8</cupRequest>
                        <availablePodCount>0</availablePodCount>
                  </AvailablePodInfoList>
            </InstanceInfo>
            <NetInfo>
                  <ProdCIDR>172.20.0.0/16</ProdCIDR>
                  <NetPlugType>Terway</NetPlugType>
                  <ServiceCIDR>172.21.0.0/20</ServiceCIDR>
            </NetInfo>
      </Result>
</QueryClusterDetailResponse>
```

`JSON` **格式**

```
{
    "RequestId": "16A78EEB-E502-4DC5-A9A0-9969980F1B58",
    "Code": 0,
    "Success": true,
    "Result": {
      "BasicInfo": {
        "ClusterInstanceId": "c6fd41efd9df74f50a8e7804fcb73c27a",
        "InstallLogInProcess": false,
        "HasInstallLogController": true,
        "Vswitchs": [
          "vsw-8vbr9ukajnfj1i98otgpn",
          "vsw-8vbq2k29mzi65lo7c6lzr",
          "vsw-8vbl8s66ry10ch52hw3u4"
        ],
        "HasInstallArmsPilot": true
```

```
        "hasInstallArmsPilot": true,
        "VpcId": "vpc-8vbm6coyrh92bbjj1ej0n",
        "RegionName": "cn-zhangjiakou",
        "InstallArmsInProcess": false,
        "ClusterName": "杭羽测试-正式环境",
        "RegionId": "cn-zhangjiakou"
      },
      "WorkLoad": {
        "MemLevel": 0,
        "AllocateAppPodCount": 17,
        "CpuRequestPercent": 0.57,
        "MemRequest": 25769803776,
        "MemUsePercent": 0.46,
        "CpuUsePercent": 0.02,
        "CpuUse": 0.69,
        "MemTotal": 56292110336,
        "CpuLevel": 0,
        "MemUse": 25901572096,
        "MemRequestPercent": 0.46,
        "AllNodeCount": 7,
        "AllocateAllPodCount": 72,
        "CpuCapacityTotal": 32,
        "MemCapacityTotal": 63632142336,
        "CpuRequest": 18.12,
        "CpuTotal": 32
      },
      "InstanceInfo": {
        "AllocatedPodInfoList": [
          {
            "appName": "image_php44",
            "envName": "正式环境",
            "memRequest": "1GB",
            "appId": 5687,
            "cupRequest": "1vcpu",
            "podCount": 1,
            "envId": 1136
          },
          {
            "appName": "image_nodejs",
            "envName": "正式环境",
            "memRequest": "512MB",
            "appId": 5691,
            "cupRequest": "0.5vcpu",
            "podCount": 1,
            "envId": 1139
          },
          {
            "appName": "image_php55",
            "envName": "正式环境",
            "memRequest": "1GB",
            "appId": 5685,
            "cupRequest": "1vcpu",
            "podCount": 1,
            "envId": 1132
          },
```

```
        },
        {
          "appName": "imgae_php72_newnew",
          "envName": "正式环境",
          "memRequest": "512MB",
          "appId": 6063,
          "cupRequest": "0.5vcpu",
          "podCount": 1,
          "envId": 1420
        },
        {
          "appName": "springboot2_prom",
          "envName": "正式环境",
          "memRequest": "1500MB",
          "appId": 6974,
          "cupRequest": "1vcpu",
          "podCount": 2,
          "envId": 1874
        },
        {
          "appName": "image_php72_new",
          "envName": "正式环境",
          "memRequest": "256MB",
          "appId": 6017,
          "cupRequest": "0.5vcpu",
          "podCount": 1,
          "envId": 1385
        },
        {
          "appName": "image_springboot",
          "envName": "正式环境",
          "memRequest": "2GB",
          "appId": 5665,
          "cupRequest": "1vcpu",
          "podCount": 1,
          "envId": 1116
        },
        {
          "appName": "imae_springboot2",
          "envName": "正式环境",
          "memRequest": "1GB",
          "appId": 5819,
          "cupRequest": "1vcpu",
          "podCount": 1,
          "envId": 1270
        },
        {
          "appName": "cms_test",
          "envName": "正式环境",
          "memRequest": "1GB",
          "appId": 5926,
          "cupRequest": "0.5vcpu",
          "podCount": 1,
          "envId": 1323
        },
```

```
        {
          "appName": "java_prom",
          "envName": "正式环境",
          "memRequest": "1500MB",
          "appId": 6945,
          "cupRequest": "1vcpu",
          "podCount": 2,
          "envId": 1863
        },
        {
          "appName": "image_php72",
          "envName": "正式环境",
          "memRequest": "512MB",
          "appId": 6015,
          "cupRequest": "0.5vcpu",
          "podCount": 1,
          "envId": 1383
        },
        {
          "appName": "cms_group",
          "envName": "正式环境",
          "memRequest": "512MB",
          "appId": 5927,
          "cupRequest": "0.5vcpu",
          "podCount": 1,
          "envId": 1325
        },
        {
          "appName": "add",
          "envName": "正式环境",
          "memRequest": "4900MB",
          "appId": 5821,
          "cupRequest": "1.5vcpu",
          "podCount": 1,
          "envId": 1273
        },
        {
          "appName": "image_nodejs4",
          "envName": "正式环境",
          "memRequest": "500MB",
          "appId": 5702,
          "cupRequest": "0.5vcpu",
          "podCount": 1,
          "envId": 1152
        },
        {
          "appName": "image_php56",
          "envName": "正式环境",
          "memRequest": "1GB",
          "appId": 5674,
          "cupRequest": "1vcpu",
          "podCount": 1,
          "envId": 1124
        }
```

```
        ],
        "AllocatePodCount": 17,
        "AppCount": 15,
        "AvailablePodInfoList": [
          {
            "memRequest": "1",
            "cupRequest": "1",
            "availablePodCount": 9
          },
          {
            "memRequest": "4",
            "cupRequest": "2",
            "availablePodCount": 4
          },
          {
            "memRequest": "8",
            "cupRequest": "4",
            "availablePodCount": 1
          },
          {
            "memRequest": "16",
            "cupRequest": "8",
            "availablePodCount": 0
          }
        ]
      },
      "NetInfo": {
        "ProdCIDR": "172.20.0.0/16",
        "NetPlugType": "Terway",
        "ServiceCIDR": "172.21.0.0/20"
      },
      "NodeWorkLoadList": []
    }
  }
```


错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

6.4.5. AddClusterNode

调用AddClusterNode接口向集群中添加ECS主机

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。

请求参数

名称	类型	是否必选	示例值	描述

Action	String	是	AddClusterNode	系统规定参数。取值：AddClusterNode。
ClusterInstanceId	String	是	ca93f56*******f0b179	集群实例id
EcsInstanceIdList.N	RepeatList	是	i-jhjhh88hhj	ECS实例id

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码，0表示成功
ErrMsg	String	添加异常	错误信息
RequestId	String	4UJF-JGH-JHJ	请求id
Success	Boolean	true	是否成功
Result	Struct		结果
Nonsense	Integer	1	该字段无意义

示例

请求示例

```
http(s)://[Endpoint]/?Action=AddClusterNode
&ClusterInstanceId=ca93f56*******f0b179
&EcsInstanceIdList.1=i-jhjhh88hhj
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<AddClusterNodeResponse>
      <RequestId>E83DFB0B-633B-4B26-9373-DDAB99AAB8C3</RequestId>
      <Code>0</Code>
      <Success>true</Success>
      <Result>
            <Nonsense>1</Nonsense>
      </Result>
</AddClusterNodeResponse>
```

`JSON` 格式

```
{
    "RequestId": "E83DFB0B-633B-4B26-9373-DDAB99AAB8C3",
    "Code": 0,
    "Success": true,
    "Result": {
      "Nonsense": 1
    }
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

6.4.6. RemoveClusterNode

调用RemoveClusterNode接口移除集群中的ECS节点

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。

请求参数

名称	类型	是否必选	示例值	描述
ClusterInstanceId	String	是	mhgh0hgg2dd	集群实例id
EcsInstanceIdList.N	RepeatList	是	i-hjhjhdd	主机ECSid

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码
ErrMsg	String	节点不在集群中	错误信息
RequestId	String	4JGH-UIUIOYKH	请求id
Success	Boolean	true	是否成功
Result	Struct		数据
Nonsense	Integer	1	

示例

请求示例

```
http(s)://[Endpoint]/?Action=RemoveClusterNode
&ClusterInstanceId=mhgh0hgg2dd
&EcsInstanceIdList.1=i-hjhjhdd
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<RemoveClusterNodeResponse>
      <RequestId>E44BDA77-942A-4898-8D5D-67797B56DB4B</RequestId>
      <Code>0</Code>
      <Success>true</Success>
      <Result>
            <Nonsense>1</Nonsense>
      </Result>
</RemoveClusterNodeResponse>
```

`JSON` 格式

```
{
    "RequestId": "E44BDA77-942A-4898-8D5D-67797B56DB4B",
    "Code": 0,
    "Success": true,
    "Result": {
      "Nonsense": 1
    }
  }
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

6.4.7. CreateCluster

调用CreateCluster接口创建K8S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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	CreateCluster	系统规定参数。取值：CreateCluster。

BusinessCode	String	是	JST	业务域： JST 聚石塔， NEW_RETAIL新零售， MINI_APP小程序云， SUPPLY供应链 MESSAGE消息业务
ClusterEnvType	String	是	TEST	集群环境类型 TEST测试 PRO正式
ClusterTitle	String	是	测试集群	集群名称
ClusterType	String	是	Standard	集群类型：Windows(windows集群) Serverless(serverless集群) Standard(标准托管版集群)
RegionId	String	是	cn-zhangjiakou	地域
VpcId	String	是	vpv-sss	vpcId
Vswitchids.N	RepeatList	是	v-sttt	vpcId下的交换机Id
CloudMonitorFlags	Integer	否	1	后面添加到集群中的主机是否安装云监控，默认为1安装
ClusterId	Long	否	0	无用 数据库主键id
CreateWithArmsIntegration	Boolean	否	true	该参数暂时无效
CreateWithLogIntegration	Boolean	否	false	是否接入日志组件，接入后会创建集群对应的SLS project。默认为false。只对标准托管版集群有效
KeyPair	String	否	test	加入集群的主机所使用的KeyPair，与password参数二选一
NetPlug	String	否	terway	该参数暂时无效
Password	String	否	Abssshh123!	加入集群的主机所使用的root密码，与keypair参数二选一;需要包括至少1个大写字母，1个小写字母，1个数字，1个特殊字符，长度至少为8

PodCIDR	String	否	172.22.0.0/16	集群中容器POD网段地址，windows集群和标准托管版集群必填
PrivateZone	Boolean	否	false	该参数暂时无效
PublicSlb	Integer	否	1	开启api server公网访问(会创建一个EIP并生成公网访问的config)
RegionName	String	否	cn-zhangjiakou	地域
ServiceCIDR	String	否	cn-zhangjiakou	集群中service的网段，windows集群和标准托管版集群必填
SnatEntry	Integer	否	1	vpc是否开启公网出口，建议开启，默认为1开启。目前的集群机器需要访问公网。如果VPC已经具备公网访问能力，或者能确保ECS都有公网IP(或EIP)，则可以设置为0

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码
ErrMsg	String	POD网段和VPC网段冲突	错误信息
RequestId	String	4HJHJ-JJJKHJKH-BJBK	请求id
Success	Boolean	true	是否成功
Result	Struct		结果数据
ClusterInstanceId	String	shjhj00	集群实例id

示例

请求示例

```
http(s)://[Endpoint]/?Action=CreateCluster
&BusinessCode=JST
&CloudMonitorFlags=1
&ClusterEnvType=TEST
&ClusterId=0
&ClusterTitle=测试集群
&ClusterType=Standard
&CreateWithArmsIntegration=true
&CreateWithLogIntegration=false
&KeyPair=test
&NetPlug=terway
&Password=Abssshh123!
&PodCIDR=172.22.0.0/16
&PrivateZone=false
&PublicSlb=1
&RegionId=cn-zhangjiakou
&RegionName=cn-zhangjiakou
&ServiceCIDR=cn-zhangjiakou
&SnatEntry=1
&VpcId=vpv-sss
&Vswitchids.1=v-sttt
&<公共请求参数>
```

正常返回示例

`XML` **格式**

```
<CreateClusterResponse>
      <RequestId>FC820A59-A46A-499D-892E-F346059D379A</RequestId>
      <Code>0</Code>
      <Success>true</Success>
      <Result>
            <ClusterInstanceId>c596149a09bfa4f90b4311f207f2c29ff</ClusterInstanceId>
      </Result>
</CreateClusterResponse>
```

`JSON` **格式**

```
{
    "RequestId": "FC820A59-A46A-499D-892E-F346059D379A",
    "Code": 0,
    "Success": true,
    "Result": {
      "ClusterInstanceId": "c596149a09bfa4f90b4311f207f2c29ff"
    }
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

6.4.8. ListAvailableClusterNode

调用ListAvailableClusterNode接口查询可添加到集群的机器列表

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	ListAvailableClusterNode	系统规定参数。取值：ListAvailableClusterNode。
ClusterInstanceId	String	是	kjhjkhk	集群实例id
PageNum	Integer	否	1	分页页码
PageSize	Integer	否	20	分页大小

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回码
ErrorMsg	String	error	错误信息
PageNumber	Integer	1	分页页码
PageSize	Integer	20	分页大小
RequestId	String	jhkh-hjhjg	请求id
TotalCount	Long	10	总数
Data	Array		列表数据
OSName	String	centos7.6	
BusinessCode	String	JST	业务域

EcsConfiguration	String	4vcpu 8G	资源配置
EcsCpu	String	4	核数
EcsEip	String	39.172.98.4	弹性公网ip
EcsExpiredTime	String	2099-12-31	过期日期
EcsLocalStorageCapacity	String	40	本地存储大小
EcsMemory	String	8192	内存
EcsOsType	String	linux	操作系统类型
EcsPrivateIp	String	192.168.49.251	私网ip
EcsPublicIp	String	39.98.115.84	公网ip
EcsZone	String	cn-zhangjiakou	可用区
InstanceId	String	i-jhjhjhj	实例id
InstanceName	String	worker	实例名称
InstanceNetworkType	String	vpc	网络类型
InstanceType	String	ecs.c6.xlarge	实例规格
InternetMaxBandwidthIn	String	1	公网入带宽最大值(单位：Mbps)
InternetMaxBandwidthOut	String	1500	公网出带宽最大值(单位：Mbps)
RegionId	String	cn-zhangjiakou	地域
VpcId	String	vpc-hjhjh	vpc

示例

请求示例

```
http(s)://[Endpoint]/?Action=ListAvailableClusterNode
&ClusterInstanceId=kjhjkhk
&<公共请求参数>
```

正常返回示例

XML 格式

```
<ListAvailableClusterNodeResponse>
      <TotalCount>5</TotalCount>
      <PageSize>20</PageSize>
      <RequestId>AB053532-71F2-43DB-B001-66545A350C4F</RequestId>
      <PageNumber>1</PageNumber>
      <Data>
            <EcsPrivateIp>192.168.48.4</EcsPrivateIp>
            <InstanceId>i-8vb1pktvmch41adzhcvl</InstanceId>
            <EcsMemory>8192</EcsMemory>
            <EcsOsType>windows</EcsOsType>
            <InstanceNetworkType>vpc</InstanceNetworkType>
            <InstanceName>worker-k8s-for-cs-
ca93f563d81de48a592dbf737a6f0b179</InstanceName>
            <InternetMaxBandwidthOut>1</InternetMaxBandwidthOut>
            <InternetMaxBandwidthIn>1500</InternetMaxBandwidthIn>
            <VpcId>vpc-8vbm6coyrh92bbjj1ej0n</VpcId>
            <EcsCpu>4</EcsCpu>
            <EcsPublicIp>39.99.154.78</EcsPublicIp>
            <EcsExpiredTime>2099-12-31</EcsExpiredTime>
            <RegionId>cn-zhangjiakou</RegionId>
            <InstanceType>ecs.c6.xlarge</InstanceType>
            <EcsZone>cn-zhangjiakou</EcsZone>
            <EcsConfiguration>4vcpu 8G</EcsConfiguration>
      </Data>
      <Data>
            <EcsPrivateIp>192.168.48.5</EcsPrivateIp>
            <InstanceId>i-8vb1pktvmch41adzhcvm</InstanceId>
            <EcsMemory>8192</EcsMemory>
            <EcsOsType>windows</EcsOsType>
            <InstanceNetworkType>vpc</InstanceNetworkType>
            <InstanceName>worker-k8s-for-cs-
ca93f563d81de48a592dbf737a6f0b179</InstanceName>
            <InternetMaxBandwidthOut>1</InternetMaxBandwidthOut>
            <InternetMaxBandwidthIn>1500</InternetMaxBandwidthIn>
            <VpcId>vpc-8vbm6coyrh92bbjj1ej0n</VpcId>
            <EcsCpu>4</EcsCpu>
            <EcsPublicIp>39.99.131.149</EcsPublicIp>
            <EcsExpiredTime>2099-12-31</EcsExpiredTime>
            <RegionId>cn-zhangjiakou</RegionId>
            <InstanceType>ecs.c6.xlarge</InstanceType>
            <EcsZone>cn-zhangjiakou</EcsZone>
            <EcsConfiguration>4vcpu 8G</EcsConfiguration>
      </Data>
      <Data>
            <EcsPrivateIp>192.168.48.8</EcsPrivateIp>
            <InstanceId>i-8vb4n7it53012zup5gjt</InstanceId>
```

```
            <EcsMemory>8192</EcsMemory>
            <EcsOsType>linux</EcsOsType>
            <InstanceNetworkType>vpc</InstanceNetworkType>
            <InstanceName>test-add101</InstanceName>
            <InternetMaxBandwidthOut>1</InternetMaxBandwidthOut>
            <InternetMaxBandwidthIn>1500</InternetMaxBandwidthIn>
            <VpcId>vpc-8vbm6coyrh92bbjj1ej0n</VpcId>
            <EcsCpu>4</EcsCpu>
            <EcsPublicIp>39.99.243.237</EcsPublicIp>
            <EcsExpiredTime>2099-12-31</EcsExpiredTime>
            <RegionId>cn-zhangjiakou</RegionId>
            <InstanceType>ecs.c6.xlarge</InstanceType>
            <EcsZone>cn-zhangjiakou</EcsZone>
            <EcsConfiguration>4vcpu 8G</EcsConfiguration>
      </Data>
      <Data>
            <EcsPrivateIp>192.168.48.2</EcsPrivateIp>
            <InstanceId>i-8vbbggdiuvh8kqlola5o</InstanceId>
            <EcsMemory>8192</EcsMemory>
            <EcsOsType>windows</EcsOsType>
            <InstanceNetworkType>vpc</InstanceNetworkType>
            <InstanceName>worker-k8s-for-cs-
cb966a639dd4b4edd85e25e673d043c1f</InstanceName>
            <InternetMaxBandwidthOut>1</InternetMaxBandwidthOut>
            <InternetMaxBandwidthIn>1500</InternetMaxBandwidthIn>
            <VpcId>vpc-8vbm6coyrh92bbjj1ej0n</VpcId>
            <EcsCpu>4</EcsCpu>
            <EcsPublicIp>39.99.149.32</EcsPublicIp>
            <EcsExpiredTime>2099-12-31</EcsExpiredTime>
            <RegionId>cn-zhangjiakou</RegionId>
            <InstanceType>ecs.c6.xlarge</InstanceType>
            <EcsZone>cn-zhangjiakou</EcsZone>
            <EcsConfiguration>4vcpu 8G</EcsConfiguration>
      </Data>
      <Data>
            <EcsPrivateIp>192.168.49.251</EcsPrivateIp>
            <InstanceId>i-8vbdtd50399yab7ihnam</InstanceId>
            <EcsMemory>8192</EcsMemory>
            <EcsOsType>windows</EcsOsType>
            <InstanceNetworkType>vpc</InstanceNetworkType>
            <InstanceName>worker-k8s-for-cs-
ca93f563d81de48a592dbf737a6f0b179</InstanceName>
            <InternetMaxBandwidthOut>1</InternetMaxBandwidthOut>
            <InternetMaxBandwidthIn>1500</InternetMaxBandwidthIn>
            <VpcId>vpc-8vbm6coyrh92bbjj1ej0n</VpcId>
            <EcsCpu>4</EcsCpu>
            <EcsPublicIp>39.98.115.84</EcsPublicIp>
            <EcsExpiredTime>2099-12-31</EcsExpiredTime>
            <RegionId>cn-zhangjiakou</RegionId>
            <InstanceType>ecs.c6.xlarge</InstanceType>
            <EcsZone>cn-zhangjiakou</EcsZone>
            <EcsConfiguration>4vcpu 8G</EcsConfiguration>
      </Data>
      <Code>0</Code>
```

```
</ListAvailableClusterNodeResponse>
```

JSON 格式


```
{
    "TotalCount": 5,
    "PageSize": 20,
    "RequestId": "AB053532-71F2-43DB-B001-66545A350C4F",
    "PageNumber": 1,
    "Data": [
      {
        "EcsPrivateIp": "192.168.48.4",
        "InstanceId": "i-8vb1pktvmch41adzhcvl",
        "EcsMemory": "8192",
        "EcsOsType": "windows",
        "InstanceNetworkType": "vpc",
        "InstanceName": "worker-k8s-for-cs-ca93f563d81de48a592dbf737a6f0b179",
        "InternetMaxBandwidthOut": "1",
        "InternetMaxBandwidthIn": "1500",
        "VpcId": "vpc-8vbm6coyrh92bbjj1ej0n",
        "EcsCpu": "4",
        "EcsPublicIp": "39.99.154.78",
        "EcsExpiredTime": "2099-12-31",
        "RegionId": "cn-zhangjiakou",
        "InstanceType": "ecs.c6.xlarge",
        "EcsZone": "cn-zhangjiakou",
        "EcsConfiguration": "4vcpu 8G"
      },
      {
        "EcsPrivateIp": "192.168.48.5",
        "InstanceId": "i-8vb1pktvmch41adzhcvm",
        "EcsMemory": "8192",
        "EcsOsType": "windows",
        "InstanceNetworkType": "vpc",
        "InstanceName": "worker-k8s-for-cs-ca93f563d81de48a592dbf737a6f0b179",
        "InternetMaxBandwidthOut": "1",
        "InternetMaxBandwidthIn": "1500",
        "VpcId": "vpc-8vbm6coyrh92bbjj1ej0n",
        "EcsCpu": "4",
        "EcsPublicIp": "39.99.131.149",
        "EcsExpiredTime": "2099-12-31",
        "RegionId": "cn-zhangjiakou",
        "InstanceType": "ecs.c6.xlarge",
        "EcsZone": "cn-zhangjiakou",
        "EcsConfiguration": "4vcpu 8G"
      },
      {
        "EcsPrivateIp": "192.168.48.8",
        "InstanceId": "i-8vb4n7it53012zup5gjt",
        "EcsMemory": "8192",
        "EcsOsType": "linux",
        "InstanceNetworkType": "vpc",
        "InstanceName": "test-add101",
        "InternetMaxBandwidthOut": "1",
```

```
        "InternetMaxBandwidthIn": "1500",
        "VpcId": "vpc-8vbm6coyrh92bbjj1ej0n",
        "EcsCpu": "4",
        "EcsPublicIp": "39.99.243.237",
        "EcsExpiredTime": "2099-12-31",
        "RegionId": "cn-zhangjiakou",
        "InstanceType": "ecs.c6.xlarge",
        "EcsZone": "cn-zhangjiakou",
        "EcsConfiguration": "4vcpu 8G"
      },
      {
        "EcsPrivateIp": "192.168.48.2",
        "InstanceId": "i-8vbbggdiuvh8kqlola5o",
        "EcsMemory": "8192",
        "EcsOsType": "windows",
        "InstanceNetworkType": "vpc",
        "InstanceName": "worker-k8s-for-cs-cb966a639dd4b4edd85e25e673d043c1f",
        "InternetMaxBandwidthOut": "1",
        "InternetMaxBandwidthIn": "1500",
        "VpcId": "vpc-8vbm6coyrh92bbjj1ej0n",
        "EcsCpu": "4",
        "EcsPublicIp": "39.99.149.32",
        "EcsExpiredTime": "2099-12-31",
        "RegionId": "cn-zhangjiakou",
        "InstanceType": "ecs.c6.xlarge",
        "EcsZone": "cn-zhangjiakou",
        "EcsConfiguration": "4vcpu 8G"
      },
      {
        "EcsPrivateIp": "192.168.49.251",
        "InstanceId": "i-8vbdtd50399yab7ihnam",
        "EcsMemory": "8192",
        "EcsOsType": "windows",
        "InstanceNetworkType": "vpc",
        "InstanceName": "worker-k8s-for-cs-ca93f563d81de48a592dbf737a6f0b179",
        "InternetMaxBandwidthOut": "1",
        "InternetMaxBandwidthIn": "1500",
        "VpcId": "vpc-8vbm6coyrh92bbjj1ej0n",
        "EcsCpu": "4",
        "EcsPublicIp": "39.98.115.84",
        "EcsExpiredTime": "2099-12-31",
        "RegionId": "cn-zhangjiakou",
        "InstanceType": "ecs.c6.xlarge",
        "EcsZone": "cn-zhangjiakou",
        "EcsConfiguration": "4vcpu 8G"
      }
    ],
    "Code": 0
  }
```


错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

7.rds数据库

7.1. DescribeDatabases

调用DescribeDatabases接口查看实例下的数据库信息。

说明：该接口暂不支持SQL Server 2017集群版、PostgreSQL、PPAS实例。

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	DescribeDatabases	系统规定参数。取值：DescribeDatabases。
InstanceId	String	是	rm-ul9wjk5xxxxxxx	实例ID

返回数据

名称	类型	示例值	描述
Code	Integer	0	响应统一code
RequestId	String	79a8ab31-32ce-4d27-a006-339a5eae3b6e	请求唯一标识
ErrMsg	String	请求失败：xxx	失败时错误信息
Result	Struct		响应数据实体
Databases	Array		数据库信息列表
DBName	String	testDBmm	数据库名称
DBStatus	String	Creating	数据库状态，取值： • Creating：创建中； • Running：使用中； • Deleting：删除中。
DBDescription	String	测试数据库	数据库描述

Engine	String	MySQL	数据库实例类型。
CharacterSetName	String	utf8	字符集，取值： MySQL/MariaDB实例：utf8、gbk、latin1、utf8mb4； SQL Server实例：Chinese_PRC_CI_AS、Chinese_PRC_CS_AS、SQL_Latin1_General_CP1_CI_AS、SQL_Latin1_General_CP1_CS_AS、Chinese_PRC_BIN。
DBInstanceId	String	rm-ul9wjk5xxxxxxx	数据库所属实例ID.
Accounts	Array		账号信息列表
AccountPrivilegeDetail	String	SELECT	账号对该数据库具有的权限。
AccountPrivilege	String	DMLOnly	账号对该数据库拥有的权限，取值： • ReadWrite：读写权限； • ReadOnly：只读权限； • DDLOnly：仅DDL权限； • DMLOnly：只DML权限。
Account	String	test	账号名称

示例

请求示例

```
http(s)://[Endpoint]/?Action=DescribeDatabases
&InstanceId=rm-ul9wjk5xxxxxxx
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<DescribeDatabasesResponse>
      <code>0</code>
   <requestId>7FE5A96C-18BF-41AF-xxxx-B85B6B4BACD7</requestId>
   <result>
      <databases>
         <accounts>
            <account>test1</account>
            <accountPrivilege>ReadOnly</accountPrivilege>
         </accounts>
         <accounts>
            <account>test2</account>
            <accountPrivilege>ReadWrite</accountPrivilege>
         </accounts>
         <characterSetName>utf8</characterSetName>
         <dBDescription></dBDescription>
         <dBInstanceId>rm-vy1xxxxxxx411rd2c</dBInstanceId>
         <dBName>sys_info</dBName>
         <dBStatus>Running</dBStatus>
         <engine>MySQL</engine>
      </databases>
   </result>
</DescribeDatabasesResponse>
```

`JSON` **格式**

```
{
 "code": 0,
 "requestId": "7FE5A96C-18BF-41AF-xxxx-B85B6B4BACD7",
 "result": {
  "databases": [{
   "accounts": [{
    "account": "test1",
    "accountPrivilege": "ReadOnly"
   }, {
    "account": "test2",
    "accountPrivilege": "ReadWrite"
   }],
   "characterSetName": "utf8",
   "dBDescription": "",
   "dBInstanceId": "rm-vy1xxxxxxx411rd2c",
   "dBName": "sys_info",
   "dBStatus": "Running",
   "engine": "MySQL"
  }]
 }
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

7.2. DeleteRdsAccount

调用DeleteRdsAccount接口删除数据库账号。

调用该接口时，实例状态需要为运行中，否则将操作失败。

说明： 该接口暂不支持SQL Server 2017集群版、PostgreSQL、PPAS实例。

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	DeleteRdsAccount	系统规定参数。取值：DeleteRdsAccount。
AccountName	String	是	test1	需要删除的数据库账号名称。
DbInstanceId	String	是	rm-uf6wjk5xxxxxxx	实例ID。

返回数据

名称	类型	示例值	描述
RequestId	String	91E855E5-7E80-4955-929B-C74EE1D38C66	请求ID。
ErrMsg	String	请求失败：xxxx	错误信息。
Code	Integer	0	统一响应码。
Result	Struct		结果
RequestId	String	91E855E5-7E80-4955-929B-C74EE1D38C66	请求ID。

示例

请求示例

```
http(s)://[Endpoint]/?Action=DeleteRdsAccount
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<DeleteRdsAccountResponse>
          <code>0</code>
          <requestId>63C7B11B-4ED8-4A29-8F21-F9609A0113BB</requestId>
          <result>
                <requestId>63C7B11B-4ED8-4A29-8F21-F9609A0113BB</requestId>
          </result>
    </DeleteRdsAccountResponse>
```

`JSON` 格式

```
{
 "code": 0,
 "requestId": "63C7B11B-4ED8-4A29-8F21-F9609A0113BB",
 "result": {
  "requestId": "63C7B11B-4ED8-4A29-8F21-F9609A0113BB"
 }
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

7.3. DescribeRdsAccounts

调用DescribeRdsAccounts接口查看实例的帐号信息。

说明 该接口暂不支持SQL Server 2017集群版、PostgreSQL、PPAS实例。

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	DescribeRdsAccounts	系统规定参数。取值：DescribeRdsAccounts。
DbInstanceId	String	是	rm-ul9wjk5xxxxxxx	实例ID
AccountName	String	否	test	数据库账号名称。

返回数据

名称	类型	示例值	描述
Code	Integer	0	响应统一code

RequestId	String	79a8ab31-32ce-4d27-a006-339a5eae3b6e	请求唯一标识。
ErrMsg	String	请求失败：xxxxx	请求失败时错误信息。
Result	Struct		结果
Accounts	Array		账号信息结果集
AccountStatus	String	Available	账号状态： • Unavailable：不可用； • Available：可用。
AccountDescription	String	测试数据库账号	账号描述。
AccountName	String	test	数据库账号名称。
AccountType	String	Normal	账号类型，取值： • Normal：普通账号； • Super：高权限账号。
PrivExceeded	String	0	账号管理的数据库是否超过最大数量限制，取值： • 1：是； • 0：否。
DBInstanceId	String	rm-ul9wjk5xxxxxxx	账号所属实例ID.
DatabasePrivileges	Array		具体权限信息
DBName	String	test	数据库名称。
AccountPrivilege	String	ReadWrite	账号的权限，取值： • ReadWrite：读写； • ReadOnly：只读； • DDLOnly：仅DDL； • DMLOnly：只DML； • Custom：自定义，您可以通过命令修改。
AccountPrivilegeDetail	String	SELECT,INSERT	账号具体的权限。

示例

请求示例

```
http(s)://[Endpoint]/?Action=DescribeRdsAccounts
&DbInstanceId=rm-ul9wjk5xxxxxxx
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<DescribeRdsAccountsResponse>
      <code>0</code>
   <requestId>7A9DDF9F-528A-xxxx-xxxx-D796096FC015</requestId>
   <result>
      <accounts>
         <accountDescription></accountDescription>
         <accountName>test1Account</accountName>
         <accountStatus>Available</accountStatus>
         <accountType>Normal</accountType>
         <dBInstanceId>rm-vy12vd8vrxxxxrd2c</dBInstanceId>
         <databasePrivileges>
            <accountPrivilege>ReadOnly</accountPrivilege>
            <accountPrivilegeDetail>SELECT,LOCK TABLES,SHOW
VIEW</accountPrivilegeDetail>
            <dBName>sys_info</dBName>
         </databasePrivileges>
         <privExceeded>0</privExceeded>
      </accounts>
      <accounts>
         <accountDescription></accountDescription>
         <accountName>lsy_admin</accountName>
         <accountStatus>Available</accountStatus>
         <accountType>Super</accountType>
         <dBInstanceId>rm-vy12vd8vrxxxxrd2c</dBInstanceId>
         <privExceeded>0</privExceeded>
      </accounts>
   </result>
</DescribeRdsAccountsResponse>
```

`JSON` 格式

```
{
 "code": 0,
 "requestId": "7A9DDF9F-528A-xxxx-xxxx-D796096FC015",
 "result": {
  "accounts": [{
   "accountDescription": "",
   "accountName": "test1Account",
   "accountStatus": "Available",
   "accountType": "Normal",
   "dBInstanceId": "rm-vy12vd8vrxxxxrd2c",
   "databasePrivileges": [{
    "accountPrivilege": "ReadOnly",
    "accountPrivilegeDetail": "SELECT,LOCK TABLES,SHOW VIEW",
    "dBName": "sys_info"
   }],
   "privExceeded": "0"
  }, {
   "accountDescription": "",
   "accountName": "lsy_admin",
   "accountStatus": "Available",
   "accountType": "Super",
   "dBInstanceId": "rm-vy12vd8vrxxxxrd2c",
   "databasePrivileges": [],
   "privExceeded": "0"
  }]
 }
}
```


错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

7.4. GrantDbToAccount

调用GrantDbToAccount接口授权账号访问数据库。

一个账号可授权访问一个或多个数据库。调用该接口时，请确保实例状态为运行中，否则将操作失败。

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	GrantDbToAccount	系统规定参数。取值：GrantDbToAccount。
AccountName	String	是	test	账号名称。

AccountPrivilege	String	是	ReadWrite	账号权限，取值： • ReadWrite：读写； • ReadOnly：只读； • DDLOnly：仅执行DDL，适用于MySQL和MariaDB； • DMLOnly：只执行DML，适用于MySQL和MariaDB； • DBOwner：数据库所有者，适用于SQL Server。
DbInstanceId	String	是	rm-ul9wjk5xxxxxxxxxx	实例ID.
DbName	String	是	testDB	需要授权访问的数据库名称。

返回数据

名称	类型	示例值	描述
Code	Integer	0	响应统一code.
RequestId	String	79a8ab31-32ce-4d27-a006-339a5eae3b6e	请求唯一标识。
ErrMsg	String	请求失败：xxxxx	请求失败时返回错误信息。

示例

请求示例

```
http(s)://[Endpoint]/?Action=GrantDbToAccount
&AccountName=test
&AccountPrivilege=ReadWrite
&DbInstanceId=rm-ul9wjk5xxxxxxxxxx
&DbName=testDB
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<GrantDbToAccountResponse>
      <code>0</code>
      <requestId>B244F851-9199-454D-922A-36478F1130FC</requestId>
</GrantDbToAccountResponse>
```

`JSON` 格式

```
{
    "code":0,
    "requestId":"B244F851-9199-454D-922A-36478F1130FC"
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

7.5. CreateDb

在某个实例下创建数据库。

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	CreateDb	系统规定参数。取值：CreateDb。
CharacterSetName	String	是	gbk	• 字符集，取值： 1. MySQL/MariaDB类型：utf8、gbk、latin1、utf8mb4; #### 1. SQLServer类型：Chinese_PRC_CI_AS、Chinese_PRC_CS_AS、SQL_Latin1_General_CP1_CI_AS、SQL_Latin1_General_CP1_CS_AS、Chinese_PRC_BIN。
DbInstanceId	String	是	rm-uf6wjk5xxxxxxxxxx	实例ID
DbName	String	是	rds_mysql	数据库名称： 说明：长度为2-64字符； 以字母开头，以字母或数字结尾； 由小写字母、数字、下划线或中划线组成； 数据库名称在实例内必须是唯一的；
DbDescription	String	否	测试用数据库	数据库描述，长度为2~256个字符。以中文、英文字母开头，可以包含数字、中文、英文、下划线（_）、短横线（-）。

返回数据

名称	类型	示例值	描述
Code	Integer	0	返回结果code
RequestId	String	07F6177E-6DE4-408A-BB4F-0723301340F3	返回请求id
ErrMsg	String	请求失败xxxx	错误信息

示例

请求示例

```
http(s)://[Endpoint]/?Action=CreateDb
&CharacterSetName=gbk
&DbInstanceId=rm-uf6wjk5xxxxxxxxxx
&DbName=rds_mysql
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<CreateDbResponse>
   <RequestId>112C0D8A-5944-4111-9FBD-0E769AFE70B3</RequestId>
   <Code>0</Code>
</CreateDbResponse>
```

`JSON` 格式

```
{
  "RequestId": "112C0D8A-5944-4111-9FBD-0E769AFE70B3",
  "Code": 0
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

7.6. DeleteDatabase

调用DeleteDatabase接口删除实例下的某个数据库 。

调用该接口时，实例必须满足以下条件，否则将操作失败：

- 实例状态为运行中；
- 实例类型为主实例；
- 数据库类型为：MySQL/SQL Server/MariaDB。

> **说明** 该接口不支持PostgreSQL、PPAS类型的实例，您需要通过SQL做DROP DATABASE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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	DeleteDatabase	系统规定参数。取值：DeleteDatabase。
DBInstanceId	String	是	rm-uf6wjk5xxxxxxx	实例ID。
DBName	String	是	testdb01	数据库名称。

返回数据

名称	类型	示例值	描述
Code	Integer	0	统一响应码。
RequestId	String	07F6177E-6DE4-408A-BB4F-0723301340F3	请求ID。
ErrMsg	String	请求失败：xxxx	错误信息。

示例

请求示例

```
http(s)://[Endpoint]/?Action=DeleteDatabase
&DBInstanceId=rm-uf6wjk5xxxxxxx
&DBName=testdb01
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<DeleteDatabaseResponse>
      <RequestId>07F6177E-6DE4-408A-BB4F-0723301340F3</RequestId>
      <Code>0</Code>
</DeleteDatabaseResponse>
```

`JSON` 格式

```
{
  "RequestId": "07F6177E-6DE4-408A-BB4F-0723301340F3",
  "Code": 0
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

7.7. GetRdsBackUp

调用GetRdsBackUp接口查看备份集列表。

备份集的状态BackupStatus必须是Success，才能用于恢复。

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	GetRdsBackUp	系统规定参数。取值：GetRdsBackUp。
DbInstanceId	String	是	rm-ul9wjk5xxxxxxx	实例ID。
BackupId	String	否	327329803	备份集ID。
BackupType	String	否	FullBackup	备份类型，取值： • FullBackup：全量备份； • IncrementalBackup：增量备份。 • 为空则表示：全量备份 & 增量备份
PageNumber	Integer	否	1	页码，取值：大于0且不超过Integer的最大值。 • 默认值：1。
PageSize	Integer	否	30	每页记录数，取值： • 30; • 50; • 100; 默认值：30.

返回数据

名称	类型	示例值	描述
Code	Integer	0	响应统一code。
ErrMsg	String	请求失败：xxxx	请求失败时返回错误信息。
RequestId	String	79a8ab31-32ce-4d27-a006-339a5eae3b6e	请求唯一标识。
Result	Struct		结果封装。
TotalRecordCount	String	100	总记录数。
PageNumber	String	1	页码。
PageRecordCount	String	30	本页备份集个数。
TotalBackupSize	Long	8672256	总备份集大小。
Items	Array		结果单例。
BackupId	String	321020562	备份集ID.
DBInstanceId	String	rm-ul9wjk5xxxxxxx	实例ID.
BackupStatus	String	Success	备份集状态。
BackupStartTime	String	2019-02-03T12:20:00Z	本次备份开始时间。格式：yyyy-MM-ddTHH:mm:ssZ（UTC时间）。
BackupEndTime	String	2019-02-13T12:20:00Z	本次备份结束时间。格式：yyyy-MM-ddTHH:mm:ssZ（UTC时间）。
BackupType	String	FullBackup	备份类型，取值： • FullBackup：全量备份； • IncrementalBackup：增量备份。
BackupMode	String	Automated	备份模式，取值： • Automated：系统自动备份； • Manual：手动备份。

BackupMethod	String	Physical	备份方式，取值： • Logical：逻辑备份； • Physical：物理备份。
BackupLocation	String	test	无
BackupExtractionStatus	String	test	无
BackupScale	String	test	无
BackupDBNames	String	spdb,sys,test20181221,test-20181228	备份数据库名称。
TotalBackupSize	Long	00000	备份集大小。
BackupSize	Long	2167808	备份文件大小，单位：Byte.
HostInstanceID	String	5882781	产生备份集的实例编号，用于区分该备份集产生于主实例或备实例。
StoreStatus	String	Disabled	该数据备份是否可删除，取值： • Enabled：可删除； • Disabled：不可删除。
MetaStatus	String	OK	库表恢复的备份集状态，取值： • OK：正常； • LARGE：表数量过多，不能用于库表恢复； • EMPTY：备份失败的备份集。 空串表示未开通库表恢复的备份集。

示例

请求示例

```
http(s)://[Endpoint]/?Action=GetRdsBackUp
&DbInstanceId=rm-ul9wjk5xxxxxxx
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<GetRdsBackUpResponse>
      <code>0</code>
   <requestId>B3CF28FF-6A0B-xxxx-9FC9-104D72380359</requestId>
   <result>
```

```
    <items>
        <backupDBNames>sys_info,test_pop</backupDBNames>
        <backupEndTime>2019-06-28 04:42:30</backupEndTime>
        <backupId>403943940</backupId>
        <backupLocation>OSS</backupLocation>
        <backupMethod>Physical</backupMethod>
        <backupMode>Automated</backupMode>
        <backupScale>DBInstance</backupScale>
        <backupSize>1875968</backupSize>
        <backupStartTime>2019-06-28 04:40:55</backupStartTime>
        <backupStatus>Success</backupStatus>
        <backupType>FullBackup</backupType>
        <dBInstanceId>rm-vy12vdxxxxx11rd2c</dBInstanceId>
        <hostInstanceID>8317197</hostInstanceID>
        <metaStatus></metaStatus>
        <storeStatus>Disabled</storeStatus>
    </items>
    <items>
        <backupDBNames>sys_info</backupDBNames>
        <backupEndTime>2019-06-26 04:42:00</backupEndTime>
        <backupId>402583673</backupId>
        <backupLocation>OSS</backupLocation>
        <backupMethod>Physical</backupMethod>
        <backupMode>Automated</backupMode>
        <backupScale>DBInstance</backupScale>
        <backupSize>1871872</backupSize>
        <backupStartTime>2019-06-26 04:40:45</backupStartTime>
        <backupStatus>Success</backupStatus>
        <backupType>FullBackup</backupType>
        <dBInstanceId>rm-vy12vxxxxx411rd2c</dBInstanceId>
        <hostInstanceID>8317197</hostInstanceID>
        <metaStatus></metaStatus>
        <storeStatus>Disabled</storeStatus>
    </items>
    <items>
        <backupDBNames>sys_info</backupDBNames>
        <backupEndTime>2019-06-25 12:06:19</backupEndTime>
        <backupId>402102345</backupId>
        <backupLocation>OSS</backupLocation>
        <backupMethod>Physical</backupMethod>
        <backupMode>Manual</backupMode>
        <backupScale>DBInstance</backupScale>
        <backupSize>1773568</backupSize>
        <backupStartTime>2019-06-25 12:05:27</backupStartTime>
        <backupStatus>Success</backupStatus>
        <backupType>FullBackup</backupType>
        <dBInstanceId>rm-vy12vxxxxx411rd2c</dBInstanceId>
        <hostInstanceID>8317197</hostInstanceID>
        <metaStatus></metaStatus>
        <storeStatus>Disabled</storeStatus>
    </items>
    <pageNumber>1</pageNumber>
    <pageRecordCount>3</pageRecordCount>
    <totalBackupSize>5521408</totalBackupSize>
```

```
      <totalRecordCount>3</totalRecordCount>
    </result>
</GetRdsBackUpResponse>
```

JSON 格式


```
{
 "code": 0,
 "requestId": "B3CF28FF-6A0B-xxxx-9FC9-104D72380359",
 "result": {
  "items": [{
   "backupDBNames": "sys_info,test_pop",
   "backupEndTime": "2019-06-28 04:42:30",
   "backupId": "403943940",
   "backupLocation": "OSS",
   "backupMethod": "Physical",
   "backupMode": "Automated",
   "backupScale": "DBInstance",
   "backupSize": 1875968,
   "backupStartTime": "2019-06-28 04:40:55",
   "backupStatus": "Success",
   "backupType": "FullBackup",
   "dBInstanceId": "rm-vy12vdxxxxx11rd2c",
   "hostInstanceID": "8317197",
   "metaStatus": "",
   "storeStatus": "Disabled"
  }, {
   "backupDBNames": "sys_info",
   "backupEndTime": "2019-06-26 04:42:00",
   "backupId": "402583673",
   "backupLocation": "OSS",
   "backupMethod": "Physical",
   "backupMode": "Automated",
   "backupScale": "DBInstance",
   "backupSize": 1871872,
   "backupStartTime": "2019-06-26 04:40:45",
   "backupStatus": "Success",
   "backupType": "FullBackup",
   "dBInstanceId": "rm-vy12vxxxxx411rd2c",
   "hostInstanceID": "8317197",
   "metaStatus": "",
   "storeStatus": "Disabled"
  }, {
   "backupDBNames": "sys_info",
   "backupEndTime": "2019-06-25 12:06:19",
   "backupId": "402102345",
   "backupLocation": "OSS",
   "backupMethod": "Physical",
   "backupMode": "Manual",
   "backupScale": "DBInstance",
   "backupSize": 1773568,
   "backupStartTime": "2019-06-25 12:05:27",
   "backupStatus": "Success",
   "backupType": "FullBackup",
```

```
    "dBInstanceId": "rm-vy12vxxxxx411rd2c",
    "hostInstanceID": "8317197",
    "metaStatus": "",
    "storeStatus": "Disabled"
   }],
   "pageNumber": "1",
   "pageRecordCount": "3",
   "totalBackupSize": 5521408,
   "totalRecordCount": "3"
  }
 }
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

7.8. CreateAccount

调用CreateAccount创建一个rds实例账号

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	CreateAccount	系统规定参数。取值：CreateAccount。
AccountName	String	是	test	数据库账号名称。 说明 名称唯一。 以字母开头，以字母或数字结尾。 由小写字母、数字或下划线组成。 长度为2~16个字符。 其他非法字符，见禁用关键字表。
AccountPassword	String	是	asdasda	数据库账号的密码。 说明 长度为8~32个字符。 由大写字母、小写字母、数字、特殊字符中的任意三种组成。 特殊字符为!@#$&%^*()_+-=

DbInstanceId	String	是	rm-k2ja0*******5qdhu1m	实例ID。
AccountType	String	否	Normal	账号类型，取值： Normal：普通账号 Super：高权限账号 默认值：Normal。

返回数据

名称	类型	示例值	描述
Code	Integer	0	统一响应code
ErrMsg	String	错误信息	错误信息
RequestId	String	FE6AC0AD-xxxx-4E67-9928-73C11923B55F	请求唯一标识

示例

请求示例

```
http(s)://[Endpoint]/?Action=CreateAccount
&DbInstanceId=rm-k2ja0*******5qdhu1m
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<CreateAccountResponse>
  <RequestId>FE6AC0AD-2DA9-4E67-9928-73C11923B55F</RequestId>
  <ErrMsg></ErrMsg>
  <Code>0</Code>
</CreateAccountResponse>
```

`JSON` 格式

```
{
  "RequestId": "FE6AC0AD-2DA9-4E67-9928-73C11923B55F",
  "ErrMsg": "",
  "Code": 0
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。

7.9. ResetAccountPassword

调用ResetAccountPassword重置账号密码

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。

请求参数

名称	类型	是否必选	示例值	描述
Action	String	是	ResetAccountPassword	系统规定参数。取值：ResetAccountPassword。
AccountName	String	是	test	账号名称。
AccountPassword	String	是	Password	新密码。 说明： 长度为8~32个字符。 由大写字母、小写字母、数字、特殊字符中的任意三种组成。 特殊字符为!@#$&%^*()_+-=
DbInstanceId	String	是	rm-k2ja****2s5qdhu1m	实例ID。

返回数据

名称	类型	示例值	描述
Code	Integer	0	统一响应码
ErrMsg	String	错误信息	错误信息
RequestId	String	FE6AC0AD-2DA9-****-9928-73C11923B55F	请求唯一标识

示例

请求示例

```
http(s)://[Endpoint]/?Action=ResetAccountPassword
&AccountName=test
&AccountPassword=Password
&DbInstanceId=rm-k2ja****2s5qdhu1m
&<公共请求参数>
```

正常返回示例

`XML` **格式**

```
<ResetAccountPassword>
  <RequestId>FE6AC0AD-2DA9-****-9928-73C11923B55F</RequestId>
  <ErrMsg></ErrMsg>
  <Code>0</Code>
</ResetAccountPassword>
```

`JSON` **格式**

```
{
  "RequestId": "FE6AC0AD-2DA9-****-9928-73C11923B55F",
  "ErrMsg": "",
  "Code": 0
}
```

错误码

访问错误中心查看更多错误码。

访问错误中心查看更多错误码。