目录

目录 1
标签管理接口（旧版） 3
说明 3
标签管理相关接口 3
数据类型 3
请求结构 3
请求结构 3
公共参数 3
公共参数 3
返回结果 4
调用成功 4
调用失败 4
签名机制 4
创建标签 6
创建标签（CreateTag） 6
描述 6
HTTP METHOD 6
请求参数 6
返回结果 6
示例（调用成功） 6
请求示例 6
响应成功返回示例 6
删除标签 6
删除标签（DeleteTag） 6
描述 6
HTTP METHOD 7
请求参数 7
返回结果 7
示例（调用成功） 7
请求示例 7
响应成功返回示例 7
获取标签列表 7
获取标签列表（ListTags） 7
描述 7
HTTP METHOD 7
请求参数 7
返回结果 7
示例（调用成功） 8
请求示例 8
响应成功返回示例 8
获取标签键列表 8
获取标签键列表（ListTagKeys） 8
描述 8
HTTP METHOD 8
请求参数 8
返回结果 8
示例（调用成功） 8
请求示例 8
响应成功返回示例 9
获取标签值列表 9

获取标签值列表（ListTagValues） 9
描述 9
HTTP METHOD 9
请求参数 9
返回结果 9
示例（调用成功） 9
请求示例 9
响应成功返回示例 9
获取用户资源列表 10
获取用户资源列表（ListResources） 10
描述 10
HTTP METHOD 10
请求参数 10
返回结果 10
示例（调用成功） 10
请求示例 10
响应成功返回示例 10
根据资源ID获取资源标签 11
根据资源ID获取资源标签（ListTagsByResourceIds） 11
描述 11
HTTP METHOD 11
请求参数 11
返回结果 11
示例（调用成功） 11
请求示例 11
响应成功返回示例 11
批量替换资源标签 12
批量替换资源标签（ReplaceResourcesTags） 12
描述 12
HTTP METHOD 12
请求参数 12
返回结果 12
示例（调用成功） 12
请求示例 12
响应成功返回示例 12
解绑资源标签 12
解绑资源标签（DetachResourceTags） 12
描述 12
HTTP METHOD 12
请求参数 13
返回结果 13
示例（调用成功） 13
请求示例 13
响应成功返回示例 13

标签管理接口（旧版）

说明

旧版（1.0）标签接口请点击此处查看。建议您升级API接口至最新版本。

标签管理相关接口

API版本 2016-03-04

- 创建标签
- 删除标签
- 描述标签
- 描述标签名称
- 描述标签分类

数据类型

- Tag
- Resource
- Filter
- TagDescription

请求结构

请求结构

客户调用金山云**标签管理**服务的OpenAPI接口是通过向指定服务地址发送请求，并按照OpenAPI文档说明在请求中添加相应的公共参数和接口参数来完成的。

访问控制OpenAPI的请求结构组成如下：

1. 服务地址

   访问控制的服务接入地址为：tagv2.api.ksyun.com

2. 通信协议

   支持通过 HTTP 或 HTTPS 两种方式进行请求通信，推荐使用安全性更高的 HTTPS方式发送请求。

3. 请求方法

   访问控制的OpenAPI同时支持GET和POST请求。

   注意

   - 不能混合使用两种请求方式。如果使用 GET 方式，参数均从 querystring 取得；如果使用 POST 方式，参数均从请求Body中取得
   - 如果请求方式是GET，需要对所有请求参数做URL编码；如果请求方式是POST，需要使用x-www-form-urlencoded方式进行编码。

4. 请求参数

   金山云OpenAPI请求包含两类参数：**公共请求参数**和**接口请求参数**。其中，公共请求参数是每个接口都要用到的请求参数，具体可参见公共参数小节；接口请求参数是各个接口所特有的，具体见各个接口的“请求参数”描述。

5. 字符编码

   请求及返回结果都使用UTF-8字符集进行编码。

公共参数

公共参数

传入方式

a. GET请求,放在url的query里面. e.g: xx.api.ksyun.com?{业务参数}&{公共参数}

b. POST请求,放在http body里面. e.g: {业务参数}&{公共参数}

参数说明

公共请求参数(必须)是每个OpenAPI接口都需要使用的请求参数。

名称	类型	是否必须参数	描述
Accesskey	String	是	ak 用户在控制台创建的accesskey
Service	String	是	服务名称，固定值tagv2
Action	String	是	操作接口名，与调用的具体openAPI相关
Version	String	是	接口版本号，固定值：2020-09-01
Timestamp	String	是	时间，UTC格式，例如：2019-08-13T17:18:36Z
SignatureVersion	String	是	签名版本号，固定值：1.0
SignatureMethod	String	是	签名算法，固定值：HMAC-SHA256
Signature	String	是	签名 签名算法
Region	String	否	区域，例如：cn-beijing-6
SecurityToken	String	否	安全令牌，在使用临时ak(STS 角色扮演获取的ak/sk),需要传该字段，如果使用GET方法，需要对该字段进行urlencode
DryRun	Boolean	否	检查当前调用者是否有权限执行相关操作，而不是真的调用执行相关操作

返回结果

调用金山云的openAPI服务，调用成功，返回的HTTP状态码（Status）为200；调用失败，返回4xx 或5xx的HTTP状态码（Status）。

金山云的**API网关**服务的调用返回的数据格式仅支持json。

调用成功

- json格式示例

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

调用失败

调用接口失败，不会返回结果数据；HTTP请求返回一个4xx或5xx的HTTP状态码，返回的HTTP消息体中包含具体的错误代码(code)及错误信息(message)；与调用成功一样还包含请求ID(RequestId)，在调用方找不到错误原因时，可以联系金山云客服，并提供RequestId，以便我们尽快帮您解决问题。

- json格式示例

```
{
    "RequestId": "68093a99-2f63-4f39-8f70-3047ab8ecb5b",
    "Error": {
        "Type": "Sender",
        "Code": "InvalidParameterValue",
        "Message": "An invalid or out-of-range value was supplied for the input parameter PathPrefix."
    }
}
```

签名机制

API网关的openAPI调用采用AWS签名算法版本4，具体可以参考AWS文档，支持GET和POST两种HTTP方法，GET方法所有请求参数包括signature放置在url中，POST方法则将Signature以名为authorization header的形式放置在header中，其主要区别在于GET方式处理的请求url长度不能过长。

签名计算的主要流程如下：

1. **创建一个正规化请求**

   在签名前，首先将请求进行正规化格式化，目的是让签名计算过程无二意，其主要过程伪代码如下：

   ```
   CanonicalRequest = HTTPRequestMethod + '\n' + CanonicalURI + '\n' + CanonicalQueryString + '\n' + CanonicalHeaders + '\n' +
    SignedHeaders + '\n' + HexEncode(Hash(RequestPayload))
   ```

   Hash指代计算哈希的算法，目前使用SHA-256，HexEncode是对哈希值进行用16进制编码（使用小写字母）。

   具体步骤如下

   1. 抽取HTTP请求方法（如GET、PUT、POST）结尾附加“换行符”
   2. URI绝对路径进行URI编码得到正规化URI，如果绝对路径为空，那么使用前斜线"/"，结尾附加“换行符”
   3. 构建正规化Querystring，结尾附加“换行符”
      - URI编码每一个querystring参数名称和参数值（注：GET方式需要包含哈希算法、信任状、签名日期和签名header等全部参数）
      - 按照ASCII字节顺序对参数名称严格排序
      - 将排序好的参数名称和参数值用=连接，按照排序结果将“参数对”用&连接
   4. 构建正规化headers，结尾附加“换行符”，伪代码如下：

      ```
      CanonicalHeaders = CanonicalHeadersEntry0 + CanonicalHeadersEntry1 + ... + CanonicalHeadersEntryN
      ```

      其中：

      ```
      CanonicalHeadersEntry = Lowercase(HeaderName) + ':' + Trimall(HeaderValue) + '\n'
      ```

      `lowercase`表示将header名字转为小写字母，`trimall`表示去掉header值前和值后的白空格，并将header值里面的连续白空格变成单空格，但是不去掉双引号中间的任何空格，且最后的正规化headers是按照header名称排序后的结果

   5. 添加签名headers，结尾附加“换行符”。签名header是包含在正规化headers中名称列表，其目的是指明哪些header参与签名计算，从而忽略请求被proxy添加的额外header，其中host、x-amz-date两个header如果存在则必须添加进来，伪代码如下

      ```
      SignedHeaders = Lowercase(HeaderName0) + ';' + Lowercase(HeaderName1) + ";" + ... + Lowercase(HeaderNameN)
      ```

      然后处理请求body，如下

   6. 对请求body使用哈希算法（SHA256）计算哈希值，并将二进制哈希值结果用16进制编码表示出来（且不使用大写字符），伪代码

      ```
      HashedPayload = Lowercase(HexEncode(Hash(requestPayload)))
      ```

      此时，

   7. 将上述i-vi步骤的结果连接成一个字符串，即为**正规化请求（Canonical Request）**
   8. 将vii步的正规化请求使用vi步的哈希算法计算哈希值

2. **创建签名字符串**

   签名字符串主要包含请求以及正规化请求的元数据信息，由签名算法、请求日期、信任状和正规化请求哈希值连接组成，伪代码如下：

   ```
   StringToSign = Algorithm + '\n' + RequestDate + '\n' + CredentialScope + '\n' + HashedCanonicalRequest
   ```

   其中，签名算法(Algorithm)为AWS4-HMAC-SHA256，请求日期(RequestDate)格式YYYYMMDD'T'HHMMSS'Z'，信任状(CredentialScope)格式为*YYYYMMDD/region/service/*aws4_request（包括请求日期（ISO 8601 基本格式）），正规化请求哈希值为上述1中第viii步的结果（注意结尾不附加“换行符”）

3. **计算签名信息**

   在计算签名前，首先从私有访问密钥（secret AccessKey）派生出签名密钥（signing key），而不是直接使用私有访问密钥；之后使用签名密钥和2中计算的签名字符串来计算签名值，具体计算过程如下

   1. 生成签名密钥，伪代码如下

      ```
      kSecret = *Your KSC Secret Access Key*
      kDate = HMAC("AWS4" + kSecret, Date)
      kRegion = HMAC(kDate, Region)
      kService = HMAC(kRegion, Service)
      kSigning = HMAC(kService, "aws4_request")
      ```

      其方式是通过HMAC算法依次生成下一个HMAC的key值（第一个为私有访问密钥字符串），而data值则依次为信任状中的各项内容（日期、region、服务、结尾字符串）；HMAC算法采用HMAC-SHA256，返回值为哈希值二进制形式（256bit，32字节），不需要做8/16进制编码显示。

2. 计算签名,伪代码如下：

```
signature = HexEncode(HMAC(derived-signing-key, string-to-sign))
```

使用HMAC-SHA256算法，以签名密钥作为key，签名字符串作为data计算签名，签名后的二进制哈希值结果以16进制编码输出。

创建标签

创建标签（CreateTag）

描述

创建标签

HTTP METHOD

GET | POST

请求参数

公共参数列表 详情

名称	类型	必填	长度限制（字符）	参数格式	描述
Action	String	是	确定值	[a-zA-Z]+	操作接口名，系统规定参数，取值：CreateTag
Version	String	是	确定值		固定为：2020-09-01
Key	String	是	1-128个字符，仅支持中英文字符、数字及+-=._/@:		标签键
Value	String	否	256个字符内，仅支持中英文字符、数字及+-=._/@()[] （） 【】：		标签值

返回结果

通过指定 Accept:application/json，返回的数据格式为json

示例（调用成功）

请求示例

```
curl -X GET|POST \
'http://tagv2.api.ksyun.com/?Action=CreateTag&Version=2020-09-01&Key=xxx&Value=xxx' \
-H 'authorization: xxx, SignedHeaders=accept;content-type;host;x-amz-date, Signature=xxx' \
-H 'accept: application/json' \
-H 'content-type: application/x-www-form-urlencoded' \
-H 'host: tagv2.api.ksyun.com' \
-H 'x-amz-date: 20200720T022802Z'
```

响应成功返回示例

```
{
    "Result": true,
    "RequestId": "7de19b5e-0791-4269-872b-c3e05b53c3c8"
}
```

访问API错误码查看更多错误码

删除标签

删除标签（DeleteTag）

描述

删除标签

HTTP METHOD

GET | POST

请求参数

公共参数列表 详情

名称	类型	必填	长度限制（字符）	参数格式	描述
Action	String	是	确定值	[a-zA-Z]+	操作接口名，系统规定参数，取值：DeleteTag
Version	String	是	确定值		固定为：2020-09-01
Tags	Array	是	数组		示例：[{"Key":"xx","Value":"xx"}]

返回结果

通过指定 Accept:application/json，返回的数据格式为json

示例（调用成功）

请求示例

```
curl -X GET|POST \
'http://tagv2.api.ksyun.com/?Action=DeleteTag&Version=2020-09-01&Tags=xxx' \
-H 'authorization: xxx, SignedHeaders=accept;content-type;host;x-amz-date, Signature=xxx' \
-H 'accept: application/json' \
-H 'content-type: application/x-www-form-urlencoded' \
-H 'host: tagv2.api.ksyun.com' \
-H 'x-amz-date: 20200720T022802Z'
```

响应成功返回示例

```
{
    "Result": true,
    "RequestId": "7de19b5e-0791-4269-872b-c3e05b53c3c8"
}
```

访问API错误码查看更多错误码

获取标签列表

获取标签列表（ListTags）

描述

获取标签列表

HTTP METHOD

GET | POST

请求参数

公共参数列表 详情

名称	类型	必填	长度限制（字符）	参数格式	描述
Action	String	是	确定值	[a-zA-Z]+	操作接口名，系统规定参数，取值：ListTags
Version	String	是	确定值		固定为：2020-09-01
Key	String	否	128个字符内		标签键
Value	String	否	256个字符内		标签值
Page	Integer	否		页码	
PageSize	Integer	否		每页数据量	

返回结果

通过指定 Accept:application/json，返回的数据格式为json

示例（调用成功）

请求示例

```
curl -X GET|POST \
'http://tagv2.api.ksyun.com/?Action=ListTags&Version=2020-09-01' \
-H 'authorization: xxx, SignedHeaders=accept;content-type;host;x-amz-date, Signature=xxx' \
-H 'accept: application/json' \
-H 'content-type: application/x-www-form-urlencoded' \
-H 'host: tagv2.api.ksyun.com' \
-H 'x-amz-date: 20200720T022802Z'
```

响应成功返回示例

```
{
    "Tags": [
        {
            "Id": 1,
            "Key": "USage",
            "Value": "",
            "CreateTime": "2020-10-10 15:24:44",
            "CanDelete": 0,
            "IsBillTag": 0
        }
    ],
    "Page": 1,
    "PageSize": 10,
    "Total": 30,
    "RequestId": "7de19b5e-0791-4269-872b-c3e05b53c3c8"
}
```

访问API错误码查看更多错误码

获取标签键列表

获取标签键列表（ListTagKeys）

描述

获取标签键列表

HTTP METHOD

GET | POST

请求参数

公共参数列表 详情

名称	类型	必填	长度限制（字符）	参数格式	描述
Action	String	是	确定值	[a-zA-Z]+	操作接口名，系统规定参数，取值：ListTagKeys
Version	String	是	确定值		固定为：2020-09-01
Page	Integer	否		页码	
PageSize	Integer	否		每页数据量	

返回结果

通过指定 Accept:application/json，返回的数据格式为json

示例（调用成功）

请求示例

```
curl -X GET|POST \
'http://tagv2.api.ksyun.com/?Action=ListTagKeys&Version=2020-09-01' \
-H 'authorization: xxx, SignedHeaders=accept;content-type;host;x-amz-date, Signature=xxx' \
-H 'accept: application/json' \
```

```
-H 'content-type: application/x-www-form-urlencoded' \
-H 'host: tagv2.api.ksyun.com' \
-H 'x-amz-date: 20200720T022802Z'
```

响应成功返回示例

```
{
    "TagKeys": [
        "USage",
        "Tag"
    ],
    "Page": 1,
    "PageSize": 0,
    "Total": 12,
    "RequestId": "7de19b5e-0791-4269-872b-c3e05b53c3c8"
}
```

访问API错误码查看更多错误码

获取标签值列表

获取标签值列表（ListTagValues）

描述

获取标签值列表

HTTP METHOD

GET | POST

请求参数

公共参数列表 详情

名称	类型	必填	长度限制（字符）	参数格式	描述
Action	String	是	确定值	[a-zA-Z]+	操作接口名，系统规定参数，取值：ListTagValues
Version	String	是	确定值		固定为：2020-09-01
TagKeys	String	是		标签键，多个用逗号连接	
Page	Integer	否		页码	
PageSize	Integer	否		每页数据量	

返回结果

通过指定 Accept:application/json，返回的数据格式为json

示例（调用成功）

请求示例

```
curl -X GET|POST \
'http://tagv2.api.ksyun.com/?Action=ListTagValues&Version=2020-09-01' \
-H 'authorization: xxx, SignedHeaders=accept;content-type;host;x-amz-date, Signature=xxx' \
-H 'accept: application/json' \
-H 'content-type: application/x-www-form-urlencoded' \
-H 'host: tagv2.api.ksyun.com' \
-H 'x-amz-date: 20200720T022802Z'
```

响应成功返回示例

```
{
    "TagValues": [
        {
            "Id": 2059,
            "Key": "TagKey1",
            "Value": "value1",
```

```
            "CreateTime": "2020-09-16 16:07:16"
        }
    ],
    "Page": 1,
    "PageSize": 0,
    "Total": 1,
    "RequestId": "7de19b5e-0791-4269-872b-c3e05b53c3c8"
}
```


访问API错误码查看更多错误码

获取用户资源列表

获取用户资源列表（ListResources）

描述

获取用户资源列表

HTTP METHOD

GET | POST

请求参数

公共参数列表 详情

名称	类型	必填	长度限制（字符）	参数格式	描述
Action	String	是	确定值	[a-zA-Z]+	操作接口名，系统规定参数，取值：ListResources
Version	String	是	确定值		固定为：2020-09-01
ResourceType	String	是	64个字符内		资源类型英文简写
ProjectIds	String	是			项目ID，多个用逗号连接，获取用户项目列表参见：获取主账号/子用户下所有项目列表
RegionCodes	String	否			regioncode，多个用逗号连接
ResourceUuids	String	否			资源ID，多个用逗号连接
TagFilters	Array	否			按标签过滤，示例:[{"Key":"k1", "Value":["v1", "v2"]}]

返回结果

通过指定 Accept:application/json，返回的数据格式为json

示例（调用成功）

请求示例

```
curl -X GET|POST \
'http://tagv2.api.ksyun.com/?Action=ListResources&Version=2020-09-01&ResourceType=eip&ProjectIds=0,1' \
-H 'authorization: xxx, SignedHeaders=accept;content-type;host;x-amz-date, Signature=xxx' \
-H 'accept: application/json' \
-H 'content-type: application/x-www-form-urlencoded' \
-H 'host: tagv2.api.ksyun.com' \
-H 'x-amz-date: 20200720T022802Z'
```

响应成功返回示例


```
{
    "Resources": [
        {
            "ResourceUuid": "xxxx",
            "Tags": [
                {
                    "resourceUuid": "xxxx",
                    "tagId": 1,
```

```
                    "tagKey": "USage",
                    "tagValue": ""
                }
            ],
            "RegionCode": "cn-shanghai-3",
            "RegionName": "上海3区(VPC)"
        }
    ],
    "Page": 1,
    "PageSize": 2,
    "Total": 1,
    "RequestId": "7de19b5e-0791-4269-872b-c3e05b53c3c8"
}
```


访问API错误码查看更多错误码

根据资源ID获取资源标签

根据资源ID获取资源标签（ListTagsByResourceIds）

描述

根据资源ID获取资源标签

HTTP METHOD

GET | POST

请求参数

公共参数列表 详情

名称	类型	必填	长度限制（字符）	参数格式	描述
Action	String	是	确定值	[a-zA-Z]+	操作接口名，系统规定参数，取值：ListTagsByResourceIds
Version	String	是	确定值		固定为：2020-09-01
ResourceType	String	是	64个字符内		资源类型英文简写
ResourceUuids	String	是		资源ID，多个用逗号连接	

返回结果

通过指定 Accept:application/json，返回的数据格式为json

示例（调用成功）

请求示例

```
curl -X GET|POST \
'http://tagv2.api.ksyun.com/?Action=ListTagsByResourceIds&Version=2020-09-01&ResourceType=eip&ResourceUuids=xxx' \
-H 'authorization: xxx, SignedHeaders=accept;content-type;host;x-amz-date, Signature=xxx' \
-H 'accept: application/json' \
-H 'content-type: application/x-www-form-urlencoded' \
-H 'host: tagv2.api.ksyun.com' \
-H 'x-amz-date: 20200720T022802Z'
```

响应成功返回示例


```
{
    "Tags": [
        {
            "ResourceUuid": "xxx",
            "TagId": 1,
            "TagKey": "Department",
            "TagValue": ""
        }
    ],
    "RequestId": "7de19b5e-0791-4269-872b-c3e05b53c3c8"
```

```
}
```

访问API错误码查看更多错误码

批量替换资源标签

批量替换资源标签（ReplaceResourcesTags）

描述

批量替换资源标签

HTTP METHOD

GET | POST

请求参数

公共参数列表 详情

名称	类型	必填	长度限制（字符）	参数格式	描述
Action	String	是	确定值	[a-zA-Z]+	操作接口名，系统规定参数，取值：ReplaceResourcesTags
Version	String	是	确定值		固定为：2020-09-01
ResourceType	String	是	64个字符内		资源类型英文简写
ReplaceTags	Array	是			示例：[{"ResourceUuids":"a,b","TagIds":"1,2"}]

返回结果

通过指定 Accept:application/json，返回的数据格式为json

示例（调用成功）

请求示例

```
curl -X GET|POST \
'http://tagv2.api.ksyun.com/?Action=ReplaceResourcesTags&Version=2020-09-01&ResourceType=eip&ReplaceTags=xxx' \
-H 'authorization: xxx, SignedHeaders=accept;content-type;host;x-amz-date, Signature=xxx' \
-H 'accept: application/json' \
-H 'content-type: application/x-www-form-urlencoded' \
-H 'host: tagv2.api.ksyun.com' \
-H 'x-amz-date: 20200720T022802Z'
```

响应成功返回示例

```
{
    "Result": true,
    "RequestId": "7de19b5e-0791-4269-872b-c3e05b53c3c8"
}
```

访问API错误码查看更多错误码

解绑资源标签

解绑资源标签（DetachResourceTags）

描述

解绑资源标签

HTTP METHOD

GET | POST

请求参数

公共参数列表 详情

名称	类型	必填	长度限制（字符）	参数格式	描述
Action	String	是	确定值	[a-zA-Z]+	操作接口名，系统规定参数，取值：DetachResourceTags
Version	String	是	确定值		固定为：2020-09-01
ResourceType	String	是	64个字符内		资源类型英文简写
ResourceUuid	String	是		资源UUID	
TagIds	String	是		标签ID，多个用逗号连接	

返回结果

通过指定 Accept:application/json，返回的数据格式为json

示例（调用成功）

请求示例

```
curl -X GET|POST \
'http://tagv2.api.ksyun.com/?Action=DetachResourceTags&Version=2020-09-01&ResourceType=eip&ResourceUuid=xxx&TagIds=1,2' \
-H 'authorization: xxx, SignedHeaders=accept;content-type;host;x-amz-date, Signature=xxx' \
-H 'accept: application/json' \
-H 'content-type: application/x-www-form-urlencoded' \
-H 'host: tagv2.api.ksyun.com' \
-H 'x-amz-date: 20200720T022802Z'
```

响应成功返回示例

```
{
    "Result": true,
    "RequestId": "7de19b5e-0791-4269-872b-c3e05b53c3c8"
}
```

访问API错误码查看更多错误码