阿里云物联网套件

服务器端API

服务器端API

当您需要对接自己的业务系统时，您可以使用服务器端API与云端互通。比如通过您的服务器控制您自己的设备等。

阿里云提供POP API方便您的系统进行对接，同时提供了几种较为普遍的语言对应的SDK，比如JAVA版、PHP版、Python版。

更新历史

发布时间	更新内容	说明
2016-01-27	初始化SDK文档，初始化生成接口版本v20160104	初始化SDK文档
2016-07-06	增加设备注册接口，API版本新增ve20160530	1，新设增备注册接口见RegistDevice；2，新版本开始产品的appkey 统一命名为productKey；3，支持用户自定义设备id

调用方式

对API接口调用是通过向API的服务端地址发送HTTP GET/POST请求，并按照接口说明在请求中加入相应请求参数来完成的；根据请求的处理情况，系统会返回处理结果。

1. 请求结构
2. 公共参数
3. 返回结果
4. 签名机制

服务地址

Iot API的服务接入地址为：iot.aliyuncs.com

通信协议

支持通过HTTP或HTTPS通道进行请求通信。为了获得更高的安全性，推荐您使用HTTPS通道发送请求。

请求方法

支持HTTP GET方法发送请求，这种方式下请求参数需要包含在请求的URL中，如果是get需要对参数进行urlencode。也可以使用POST。

请求参数

每个请求都需要指定要执行的操作，即Action参数（例如Sub），以及每个操作都需要包含的公共请求参数和指定操作所特有的请求参数。

字符编码

请求及返回结果都使用UTF-8字符集进行编码。

公共请求参数

公共请求参数是指每个接口都需要使用到的请求参数。

名称	类型	是否必须	描述
Format	String	否	返回值的类型，支持JSON与XML。默认为XML
Version	String	是	API版本号，为日期形式：YYYY-MM-DD，最新版本为2016-05-30 ，每个接口可以存在多个版本
AccessKeyId	String	是	阿里云颁发给用户的访问服务所用的密钥ID
Signature	String	是	签名结果串，关于签名的计算方法，请参见签名机制。
SignatureMethod	String	是	签名方式，目前支持HMAC-SHA1
Timestamp	String	是	请求的时间戳。日期格式按照ISO8601标准表示，并需要使用UTC时间。格式为

			YYYY-MM-DDThh:mm:ssZ 例如，2016-01-04T12:00:00Z（为北京时间2016年1月04日20点0分0秒）
SignatureVersion	String	是	签名算法版本，目前版本是1.0
SignatureNonce	String	是	唯一随机数，用于防止网络重放攻击。用户在不同请求间要使用不同的随机数值
RegionId	String	是	取值 cn-hangzhou

示例

```
https://iot.aliyuncs.com/
?Format=xml
&Version=2016-05-30
&Signature=Pc5WB8gokVn0xfeu%2FZV%2BiNM1dgI%3D
&SignatureMethod=HMAC-SHA1
&SignatureNonce=15215528852396
&SignatureVersion=1.0
&AccessKeyId=key-test
&Timestamp=2012-06-01T12:00:00Z
&RegionId=cn-hangzhou
```

公共返回参数

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

示例

XML示例

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

JSON示例

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

调用API服务后返回数据采用统一格式，返回的HTTP状态码为2xx，代表调用成功；返回4xx或5xx的HTTP状态码代表调用失败。调用成功返回的数据格式主要有XML和JSON两种，外部系统可以在请求时传入参数来制定返回的数据格式，默认为XML格式。本文档中的返回示例为了便于用户查看，做了格式化处理，实际返回结果是没有进行换行、缩进等处理的。

成功结果

XML示例

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

JSON示例

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

错误结果

调用接口出错后，将不会返回结果数据。调用方可根据每个接口对应的错误码以及下述2.3.3的公共错误码来定位错误原因。

当调用出错时，HTTP请求返回一个4xx或5xx的HTTP状态码。返回的消息体中是具体的错误代码及错误信息。另外还包含一个全局唯一的请求ID：RequestId和一个您该次请求访问的站点ID：HostId。在调用方找不到错误原因时，可以联系阿里云客服，并提供该HostId和RequestId，以便我们尽快帮您解决问题。

XML示例

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

```
  <RequestId>8906582E-6722-409A-A6C4-0E7863B733A5</RequestId>
  <HostId>iot.aliyuncs.com</HostId>
  <Code>UnsupportedOperation</Code>
  <Message>The specified action is not supported.</Message>
</Error>
```

JSON示例

```
{
  "RequestId": "8906582E-6722-409A-A6C4-0E7863B733A5",
  "HostId": "iot.aliyuncs.com",
  "Code": "UnsupportedOperation",
  "Message": "The specified action is not supported."
}
```

公共错误码

IOT服务会对每个访问的请求进行身份验证，所以无论使用HTTP还是HTTPS协议提交请求，都需要在请求中包含签名（Signature）信息。 IOT通过使用Access Key ID和Access Key Secret进行对称加密的方法来验证请求的发送者身份。
Access Key ID和Access Key Secret由阿里云官方颁发给访问者（可以通过阿里云官方网站申请和管理），
其中Access Key ID用于标识访问者的身份；
Access Key Secret是用于加密签名字符串和服务器端验证签名字符串的密钥，必须严格保密，只有阿里云和用户知道。

用户在访问时，按照下面的方法对请求进行签名处理：

使用请求参数构造规范化的请求字符串（Canonicalized Query String）

按照参数名称的字典顺序对请求中所有的请求参数（包括文档中描述的“公共请求参数”和给定了的请求接口的自定义参数，但不能包括“公共请求参数”中提到Signature参数本身）进行排序。 注：当使用GET方法提交请求时，这些参数就是请求URI中的参数部分（即URI中“?”之后由“&”连接的部分）。

对每个请求参数的名称和值进行编码。名称和值要使用UTF-8字符集进行URL编码，URL编码的编码规则是：

- 对于字符 A-Z、a-z、0-9以及字符“-”、“_”、“.”、“~”不编码；
- 对于其他字符编码成“%XY”的格式，其中XY是字符对应ASCII码的16进制表示。比如英文的双引号(")对应的编码就是%22
- 对于扩展的UTF-8字符，编码成“%XY%ZA…”的格式；
- 需要说明的是英文空格()要被编码是%20，而不是加号（+）。
  注意: 一般支持URL编码的库（比如Java中的java.net.URLEncoder）都是按照

“application/x-www-form-urlencoded”的MIME类型的规则进行编码的。实现时可以直接使用这类方式进行编码，把编码后的字符串中加号（+）替换成%20、星号（*）替换成%2A、%7E替换回波浪号（~），即可得到上述规则描述的编码字符串。

对编码后的参数名称和值使用英文等号（=）进行连接。

- 再把英文等号连接得到的字符串按参数名称的字典顺序依次使用&符号连接，即得到规范化请求字符串

使用上一步构造的规范化字符串按照下面的规则构造用于计算签名的字符串：
StringToSign= HTTPMethod +"&"+ percentEncode("/) + "&" + percentEncode(CanonicalizedQueryString)
其中HTTPMethod是提交请求用的HTTP方法，比GET。
percentEncode(“/”)是按照1.b中描述的URL编码规则对字符“/”进行编码得到的值，即"%2F".
percentEncode(CanonicalizedQueryString)是对第1步中构造的规范化请求字符串按1.b中描述 的URL编码规则编码后得到的字符串。

3. 按照RFC2104的定义，使用上面的用于签名的字符串计算签名HMAC值。注意：计算签名时使用的Key就是用户持有的Access Key Secret并加上一个“&”字符(ASCII:38)，使用的哈希算法是SHA1。
4. 按照Base64编码规则把上面的HMAC值编码成字符串，即得到签名值（Signature）。
5. 将得到的签名值作为Signature参数添加到请求参数中，即完成对请求签名的过程。 注意：得到的签名值在作为最后的请求参数值提交给服务器的时候，要和其他参数一样，按照RFC3986的规则进行URL编码）。

以Sub为例，签名前的请求URL为: http://iot.aliyuncs.com/?Format=XML&SignatureMethod=HMAC-SHA1&Topic.1=%2F60027911%2Ftopic1&Timestamp=2016-05-05T03%3A03%3A28Z&Action=Sub&AccessKeyId=testId&SubCallback=http%3A%2F%2Flocalhost%3A18080%2Fmock%2Fconsumer&RegionId=cn-hangzhou&SignatureNonce=947519ce-68ee-4546-8508-69e0338d3568&AppKey=123&Version=2016-01-04&SignatureVersion=1.0

那么StringToSign就是：

GET&%2F&AccessKeyId%3DtestId%26Action%3DSub%26AppKey%3D123%26Format%3DXML%26RegionId%3Dcn-hangzhou%26SignatureMethod%3DHMAC-SHA1%26SignatureNonce%3D085c2857-4cbf-4fe2-82ca-2ac00062c7d4%26SignatureVersion%3D1.0%26SubCallback%3Dhttp%253A%252F%252Flocalhost%253A18080%252Fmock%252Fconsumer%26Timestamp%3D2016-05-05T03%253A02%253A18Z%26Topic.1%3D%252F60027911%252Ftopic1%26Version%3D2016-01-04

假如使用的Access Key Id是“testId”，Access Key Secret是“test”，用于计算HMAC的Key就是“test&”，则计算得到的签名值是： vBz5BwUdebR0lGtrLySmjRv%2Fizs%3D
签名后的请求URL为（注意增加了Signature参数）：

http://iot.aliyuncs.com/?Format=XML&SignatureMethod=HMAC-SHA1&Topic.1=%2F60027911%2Ftopic1&Signature=vBz5BwUdebR0lGtrLySmjRv%2Fizs%3D&Timest

amp=2016-05-05T03%3A03%3A28Z&Action=Sub&AccessKeyId=testId&SubCallback=http%3A%2F%2Flocalhost%3A18080%2Fmock%2Fconsumer&RegionId=cn-hangzhou&SignatureNonce=947519ce-68ee-4546-8508-69e0338d3568&AppKey=123&Version=2016-01-04&SignatureVersion=1.0

- JAVA SDK
- .Net SDK
- Php SDK
- Python SDK

版本更迭

2016-08-15

- 升级SDK版本到2.0.3
- 增加服务端订阅者上线API
- 增加Publish消息Qos特性

2015-01-26

- 增加取消订阅topic接口
- 增加推送消息至设备接口
- 增加推送消息并获取设备结果接口
- 增加针对Topic或者设备的授权接口

SDK使用参考

公共参数

名称	类型	是否必须	描述
accessKeyId	String	必须	阿里云的Access Key ID
accessKeySecret	String	必须	阿里云的Access Key Secret

- accessKeyId和accessKeySecret 在阿里云官网控制台获取 https://ak-console.aliyun.com/#/accesskey
- 各个方法中需要用到 ProductKey 在IoT控制台 查看ProductKey数据.

引入SDK

*1 maven坐标：

```
<!-- https://mvnrepository.com/artifact/com.aliyun/aliyun-java-sdk-iot -->
<dependency>
<groupId>com.aliyun</groupId>
<artifactId>aliyun-java-sdk-iot</artifactId>
<version>2.0.3</version>
</dependency>
```

依赖公共包：

```
<dependency>
<groupId>com.aliyun</groupId>
<artifactId>aliyun-java-sdk-core</artifactId>
<version>2.4.2</version>
</dependency>
```

*2 sdk源码地址参考 - github https://github.com/aliyun/aliyun-openapi-java-sdk/tree/master/aliyun-java-sdk-iot

*3 其它下载方式

- Java Core SDK Jar
- Java SDK Jar

初始化

```
String accessKey = "<your accessKey>";
String accessSecret = "<your accessSecret>";
IClientProfile profile = DefaultProfile.getProfile("cn-hangzhou",accessKey, accessSecret);
DefaultAcsClient client = new DefaultAcsClient(profile);//初始化SDK客户端
```

发起调用

以推送数据到设备为例

```
RevertRpcRequest rpcRequest = new RevertRpcRequest();
rpcRequest.setDeviceName("11a936267d2a4b6eb7b4cb8549fc1fa7");//设备接入时候得到ID
rpcRequest.setProductKey(ProductKey);//设备接入时候填写的ProductKey
```

```
rpcRequest.setTimeOut(5000); //超时时间，单位毫秒.如果超过这个时间设备没反应则返回"TIMEOUT"
rpcRequest.setRpcContent("aGVsbG8gd29ybGQ=");//推送给设备的数据.数据要求二进制数据做一次BASE64编码.(示例里面是"helloworld"编码后的值)
RevertRpcResponse rpcResponse = client.getAcsResponse(rpcRequest);
System.out.println(rpcResponse.getResponseContent());//得到设备返回的数据信息.
System.out.println(rpcResponse.getRpcCode());//对应的响应码( TIMEOUT/SUCCESS/OFFLINE等)
```

公共参数

名称	类型	是否必须	描述
accessKeyId	String	必须	阿里云的Access Key ID
accessKeySecret	String	必须	阿里云的Access Key Secret

- accessKeyId和accessKeySecret 在阿里云官网控制台获取 https://ak-console.aliyun.com/#/accesskey
- 各个方法中需要用到 ProductKey 在IoT控制台 查看ProductKey数据.

引入PHP SDK

- 下载PHP CORE SDK 代码 PHP-SDK
- 在PHP文件中包含其头文件（注意文件路径要正确），如：

```
<?php
include_once '../aliyun-php-sdk-core/Config.php';
//Do something below...
快速入门
```

SDK调用示例1、实例化一个请求类,根据调用API的不同版本实例化相应的请求类2、给请求实例赋值3、调用接口返回调用结果

初始化

初始化一个AliyunClient

- 与阿里云SDK相关的文件都在aliyun文件夹下
- AliyunClient是与API交互的接口，SDK的操作都是通过AliyunClient完成的
- AliyunClient可以复用，建议设置成应用程序全局唯一的
- 用户可以修改类AliyunClient里的变量 $connectTimeout和$readTimeout来设置SDK调用接口的连接超时时间和读取超时时间，SDK默认的连接超时时间是3秒，读取超时时间是80秒

```
<?php
include_once 'aliyun-php-sdk-core/Config.php';
use \Iot\Request\v20160530 as Iot;
// 设置你的AccessKeyId/AccessSecret/ProductKey
$accessKeyId = "";
$accessSecret = "";
$iClientProfile = DefaultProfile::getProfile("cn-hangzhou", $accessKeyId, $accessSecret);
$client = new DefaultAcsClient($iClientProfile);
```

发起调用

以publish数据到设备为例:

```
$request = new Iot\PubRequest();
$request->setProductKey(123456);
$request->setMessageContent("aGVsbG93b3JsZA==");// Hello world base64 String.
$request->setTopicFullName("/60027911/home/admin/adfadsfa/dsafsfa");//消息发送给哪个topic中.
$response = $client->getAcsResponse($request);
print_r("\r\n");
print_r($response);
```

公共参数

名称	类型	是否必须	描述
accessKeyId	String	必须	阿里云的Access Key ID
accessKeySecret	String	必须	阿里云的Access Key Secret

- accessKeyId和accessKeySecret 在阿里云官网控制台获取 https://ak-console.aliyun.com/#/accesskey
- 各个方法中需要用到 ProductKey 在IoT控制台 查看ProductKey数据.

引入Python SDK

安装Python SDK包：

```
sudo pip install aliyun-python-sdk-iot
```

在Python文件中引入Python SDK相关文件：

```
from aliyunsdkcore import client
from aliyunsdkiot.request.v20160530 import RegistDeviceRequest
from aliyunsdkiot.request.v20160530 import DeviceGrantRequest
from aliyunsdkiot.request.v20160530 import DevicePermitModifyRequest
from aliyunsdkiot.request.v20160530 import ListDevicePermitsRequest
from aliyunsdkiot.request.v20160530 import DeviceRevokeByIdRequest
from aliyunsdkiot.request.v20160530 import DeviceRevokeByTopicRequest
from aliyunsdkiot.request.v20160530 import SubRequest
from aliyunsdkiot.request.v20160530 import PubRequest
from aliyunsdkiot.request.v20160530 import UnSubRequest
from aliyunsdkiot.request.v20160530 import RevertRpcRequest
```

SDK调用示例

1. 实例化一个请求类，根据调用API的不同版本实例化相应的请求类
2. 给请求实例赋值
3. 调用接口返回调用结果

初始化

```
accessKeyId = '<your accessKey>'
accessKeySecret = '<your accessSecret>'
clt = client.AcsClient(accessKeyId, accessKeySecret, 'cn-hangzhou')
```

发起调用

以publish数据到设备为例:

```
request = PubRequest.PubRequest()
request.set_accept_format('json') // 设置返回数据格式，默认为XML
request.set_ProductKey('123456')
request.set_TopicFullName('/123456/test') // 消息发送给哪个topic中
request.set_MessageContent('aGVsbG93b3JsZA==') // Hello world base64 String
result = clt.do_action(request)
print 'publish : ' + result
```

接口列表

发消息到指定的Topic

描述

请求参数

名称	类型	是否必须	描述
<公共参数>			见 公共参数
ProductKey	String	是	以哪个ProductKey来进行发送
MessageContent	String	是	发送的消息，将消息内容二进制进行BASE64转码后得到的字符串
TopicFullName	String	是	消息发送目的Topic名字,如/ProductKey/topic1/xxx
Qos	Integer	否	指定消息发送的方式;0:最多发送一次, 1: 最少发送一次 需要注意的是: 消息在IOT套件中最多保存7天

返回参数

名称	类型	描述
RequestId	String	表示调用返回码(UNKNOW,SUCCESS,TIMEOUT)
Success	Boolean	表示调用成功与否
ErrorMessage	String	出错信息

示例

- 请求示例

```
https://iot.aliyuncs.com/?&Action=Pub
&ProductKey=...
```

```
&TopicFullName=%252F1231231%252Ftopic1%252F1
&MessageContent=aGVsbG93b3JsZA
&<[公共请求参数]>
```

SDK示例代码 [SDK下载]

- java

```
PubRequest pub = new PubRequest();
pub.setProductKey(ProductKey);
pub.setMessageContent("aGVsbG93b3JsZA==");// Hello world base64 String.
pub.setTopicFullName("/.../home/admin/adfadsfa/dsafsfa");//消息发送给哪个topic中.
pub.setQos(1);//设置Qos为1，那么设备如果不在线，重新上线会收到离线消息，消息最多在Iot Hub中保存7天.
PubResponse response = client.getAcsResponse(pub);
System.out.println(response.getRequestId());//当次请求的ID
System.out.println(response.getSuccess());//请求是否成功.
System.out.println(response.getErrorMessage());//出错时的错误信息
```

php

```
$request = new Iot\PubRequest();
$request->setProductKey($productKey);
$request->setQos(0);
$request->setMessageContent("aGVsbG93b3JsZA==");// Hello world base64 String.
$request->setTopicFullName("/60027911/home/admin/adfadsfa/dsafsfa");//消息发送给哪个topic中.
$response = $client->getAcsResponse($request);
print_r("\r\n");
print_r($response);
```

python

```
request = PubRequest.PubRequest()
request.set_accept_format('json') // 设置返回数据格式，默认为XML
request.set_ProductKey('...')
request.set_TopicFullName('/.../test') // 消息发送给哪个topic中
request.set_MessageContent('aGVsbG93b3JsZA==') // Hello world base64 String
result = clt.do_action(request)
print 'publish : ' + result
```

返回示例

json示例

```
{
```

```
"RequestId":"BB71E443-4447-4024-A000-EDE09922891E",
"Success":true,
}
```

XML示例

```
<PubResponse>
<RequestId>BB71E443-4447-4024-A000-EDE09922891E</RequestId>
<Success>true</Success>
</PubResponse>
```

创建产品

描述

本接口为 2016-05-30版本新发布接口

请求参数

名称	类型	是否必须	描述
<公共参数>			见 公共参数
Name	String	是	产品名称
CatId	Long	是	产品类型ID，可通过GetCats查询
SecurityPolicy	String	否	安全策略 P_D产品和设备校验、P产品校验、D设备校验，默认为P_D
Desc	String	否	产品描述

返回参数

名称	类型	描述
RequestId	String	表示调用返回码
Success	Boolean	表示调用成功与否
ErrorMessage	String	出错信息
ProductInfo	ProductInfo	成功时返回的新建产品信息，具体见ProductInfo定义

ProductInfo定义：

名称	类型	描述
ProductName	String	产品名称
ProductKey	String	ProductKey，全局唯一
CatId	Long	产品类型ID
CreateUserId	Long	创建该产品的用户ID
ProductDesc	String	产品描述
ProductSecret	String	产品密钥
GmtCreate	String	创建时间
GmtModified	String	修改时间

示例

请求示例

```
 https://iot.aliyuncs.com/?Action=CreateProduct
&Name=TestProduct
&CatId=10000
&Desc=Create+Product+test
&<[公共请求参数]>
```

SDK示例代码 [SDK下载]

java

```
CreateProductRequest request = new CreateProductRequest();
request.setCatId(10000L);
request.setDesc("Create Product test");
request.setName("TestProduct");
CreateProductResponse response = (CreateProductResponse)executePop(request);
if(response != null){
System.out.println("Response requestId:"+response.getRequestId()+"
isSuccess:"+response.getSuccess() +" Error:"+response.getErrorMessage());
}
```

php

```
$request = new Iot\CreateProductRequest();
$request->setCatId(10000L);
```

```
$request->setDesc("Create Product test");
$request->setName("TestProduct");
$response = $client->getAcsResponse($request);
print_r("\r\n");
print_r($response);
```

python

```
request = CreateProductRequest.CreateProductRequest()
request.set_accept_format('json')
request.set_CatId(10000L)
request.set_Desc('Create Product test')
request.set_Name('TestProduct')
result = clt.do_action(request)
print 'publish : ' + result
```

返回示例

json示例

```
 {
RequestId:8AE93DAB-958F-49BD-BE45-41353C6DEBCE,
Success:true,
ProductInfo:{
ProductKey:...,
CatId:56000,
ProductName:工业产品
}
}
```

XML示例

```
 <CreateProductResponse>
<RequestId>70B70922-67BD-4506-ABFC-E99A2972809E</RequestId>
<ProductInfo>
<ProductKey>...</ProductKey>
<ProductDesc>Create Product test</ProductDesc>
<CatId>10000</CatId>
<ProductName>TestProductx</ProductName>
</ProductInfo>
<Success>true</Success>
</CreateProductResponse>
```

修改产品信息
描述

本接口为 2016-05-30版本新发布接口

请求参数

名称	类型	是否必须	描述
<公共参数>			见 公共参数
ProductKey	String	是	ProductKey作为需要更新产品的ID
CatId	Long	否	产品类型ID
ProductName	String	否	产品名称
ProductDesc	String	否	产品描述

返回参数

名称	类型	描述
RequestId	String	表示调用返回码
Success	Boolean	表示调用成功与否
ErrorMessage	String	出错信息

示例

请求示例

```
 http://iot.aliyuncs.com/?Action=UpdateProduct
&ProductKey=...
&ProductName=TestProductNew
&<[公共请求参数]>
```

SDK示例代码 [SDK下载]

java

```
UpdateProductRequest request = new UpdateProductRequest();
request.setProductKey("...");
request.setProductName("TestProductNew");
UpdateProductResponse response = (UpdateProductResponse) executePop(request);
if(response != null){
System.out.println("Response requestId:"+response.getRequestId()+"
```

```
isSuccess:"+response.getSuccess() +" Error:"+response.getErrorMessage());
}
```

php

```
$request = new Iot\UpdateProductRequest();
$request->setProductKey("...");
$request->setProductName("TestProductNew");
$response = $client->getAcsResponse($request);
print_r("\r\n");
print_r($response);
```

python

```
request = UpdateProductRequest.UpdateProductRequest()
request.set_accept_format('json')
request.set_ProductKey('...')
request.set_ProductName('TestProductNew')
result = clt.do_action(request)
print 'publish : ' + result
```

返回示例

json示例

```
 {
"RequestId":"C4FDA54C-4201-487F-92E9-022F42387458",
"Success":true,
}
```

XML示例

```
 <UpdateProductResponse>
<RequestId>C4FDA54C-4201-487F-92E9-022F42387458</RequestId>
<Success>true</Success>
</UpdateProductResponse>
```

描述

请求参数

名称	类型	是否必须	描述

<公共参数>			见 公共参数
Action	String	是	API名称,取值为 ListDevicePermits
ProductKey	Long	是	设备的ProductKey
DeviceName	String	是	设备名称

返回参数

名称	类型	描述
Success	Boolean	表示调用成功与否
RequestId	String	当前请求在阿里云产生的请求ID
devicePermissions	List	权限列表

示例

- 请求示例

```
https://iot.aliyuncs.com/?&Action=ListDevicePermits
&ProductKey=60028255
&DeviceName=11a936267d2a4b6eb7b4cb8549fc1fa7
&<[公共请求参数]>
```

SDK示例代码 [SDK下载]

- java

```
ListDevicePermitsRequest listDevicePermitsRequest = new ListDevicePermitsRequest();
listDevicePermitsRequest.setDeviceName("12cc04f3d13244658b29347cb9883008");
listDevicePermitsRequest.setProductKey(60028188l);
ListDevicePermitsResponse devicePermitsResponse =
client.getAcsResponse(listDevicePermitsRequest);
List<ListDevicePermitsResponse.DevicePermission> devicePermissions = devicePermitsResponse
.getDevicePermissions();
for (ListDevicePermitsResponse.DevicePermission devicePermission : devicePermissions) {
System.out.println(devicePermission.getTopicFullName() + ":" + devicePermission.getGrantType()
+ ":"
+ devicePermission.getTopicUserId() + ",ID:" + devicePermission.getId());
}
```

php

python

```
request = ListDevicePermitsRequest.ListDevicePermitsRequest()
request.set_accept_format('json') // 设置返回数据格式，默认为XML
request.set_ProductKey('123456')
request.set_DeviceName('xxxxxxx')
result = clt.do_action(request)
print 'list device permits : ' + result
```

返回示例

json示例

```
{
"RequestId":"120F5EB3-7023-4F0C-B419-9303AB336885",
"Success":true
"devicePermissions":[...]
}
```

XML示例

```
<ListDevicePermitsResponse>
<RequestId>82285BAF-640C-4EC2-9264-B96A27688AB7</RequestId>
<Success>true</Success>
<devicePermissions>...</devicePermissions>
</ListDevicePermitsResponse>
```

描述

本接口为 2016-05-30版本新发布接口。使用场景：用户通过服务器端生成设备。目前只支持单个设备生成。

请求参数

名称	类型	是否必须	描述
<公共参数>			见 公共参数
Version			API版本号，取值2016-05-30
Action	String	是	API名称,取值为RegistDevice
ProductKey	String	是	产品的product AppKey
DeviceName	String	否	自定义设备名称，不传

			则由系统生成默认与deviceId一致

返回参数

名称	类型	描述
Success	Boolean	表示调用成功与否
RequestId	String	当前请求在阿里云产生的请求ID
ErrorMessage	String	错误信息
DeviceId	String	阿里云颁发的设备id，全局唯一
DeviceName	String	设备名称，产品内唯一，由用户自定义，如果重复将返回已有产品，如果不指定则返回deviceId
DeviceSecret	String	设备私钥
DeviceStatus	String	设备状态，目前预留

示例

- 请求示例

```
https://iot.aliyuncs.com/?&Action=RegistDevice
&ProductKey=60028255
&<[公共请求参数]>
```

SDK示例代码 [SDK下载]

java

```
RegistDeviceRequest request = new RegistDeviceRequest();
request.setProductKey("xxxxxx");
request.setDeviceName("xxxxx");//可以设空，如果名称为空则由阿里云生成设备名称默认与设备id一致。设备名称在产品内唯一，如果已存在则返回已有设备
RegistDeviceResponse resp = client.getAcsResponse(request);

System.out.println(resp.getSuccess());
System.out.println(resp.getErrorMessage());
System.out.println(resp.getDeviceSecret());
System.out.println(resp.getDeviceId());
System.out.println(resp.getDeviceName());
```

php

python

```
request = RegistDeviceRequest.RegistDeviceRequest()
request.set_accept_format('json') // 设置返回数据格式，默认为XML
request.set_ProductKey('123456')
request.set_DeviceName('xxxxxxx') // 可以设空，如果名称为空则由阿里云生成设备名称默认与设备id一致。设备名称在产品内唯一，如果已存在则返回已有设备
result = clt.do_action(request)
print 'regist device : ' + result
```

返回示例

json示例

```
{
"RequestId":"120F5EB3-7023-4F0C-B419-9303AB336885",
"Success":true
"DeviceId":"", //阿里云颁发的设备id 全局唯一
"DeviceName":"",//设备名称，用户自定义或系统生成
"DeviceSecret":"",//设备私钥
"DeviceStatus":"",//预留状态字段
"ErrorMessage":""//错误信息
}
```

常见错误

未授权操作

这种错误信息一般是由于 调用SDK时候，填写的accessKeyId和accessSecret与你指定的productKey的创建者不是用一个用户导致的。 注册设备的时候，需要使用当前ProductKey的创建者对应的accessKey和accessSecret才有效.

特别注意：该接口目前只适用CCP协议接入的设备，MQTT协议不支持

描述

请求参数

名称	类型	是否必须	描述
<公共参数>			见 公共参数

Action	String	是	API名称,取值为RevertRpc
ProductKey	String	是	以哪个ProductKey来进行发送
DeviceName	String	是	设备名称
RpcContent	String	是	需要传递给设备的数据，数据需要为 二进制经过Base64转码得到的字符串数据
TimeOut	Integer	是	表示等待设备回复消息的时间，单位是毫秒

返回参数

名称	类型	描述
Rpccode	String	表示调用返回码(UNKNOW,SUCCESS,TIMEOUT,HALFCONN)
Success	Boolean	表示调用成功与否
ResponseContent	String	设备返回二进制数据的base64编码后的值

```
UNKNOW:系统异常
SUCCESS:成功
TIMEOUT:设备回执超时
OFFLINE: 设备离线
HALFCONN: 设备离线(设备连接断开但是断开时间未超过一个心跳周期)
```

示例

- 请求示例

```
 https://iot.aliyuncs.com/?&Action=RevertRpc
&ProductKey=60028255
&DeviceName=11a936267d2a4b6eb7b4cb8549fc1fa7
&RpcContent=aGVsbG8gd29ybGQ%253D
&TimeOut=5000
&<[公共请求参数]>
```

SDK示例代码 [SDK下载]

- java

```
RevertRpcRequest rpcRequest = new RevertRpcRequest();
rpcRequest.setDeviceName("11a936267d2a4b6eb7b4cb8549fc1fa7");//设备名称
rpcRequest.setProductKey(60028255);//设备接入时候填写的productKey
rpcRequest.setTimeOut(5000); //超时时间，单位毫秒.如果超过这个时间设备没反应则返回"TIMEOUT"
rpcRequest.setRpcContent("aGVsbG8gd29ybGQ=");//推送给设备的数据.数据要求二进制数据做一次BASE64编码.(示例里面是"helloworld"编码后的值)
RevertRpcResponse rpcResponse = client.getAcsResponse(rpcRequest);
//得到设备返回的数据信息. 注意:得到的数据是设备返回二进制数据然后再经过IOT平台base64转换之后的字符串.需要转换一下才能拿到原始的二进制数据.
System.out.println(rpcResponse.getResponseContent());
System.out.println(rpcResponse.getRpccode());//对应的响应码(TIMEOUT/SUCCESS/OFFLINE/HALFCONN等)
```

- php

```
$request = new Iot\RevertRpcRequest();
$request->setProductKey(60028255);
$request->setDeviceName("11a936267d2a4b6eb7b4cb8549fc1fa7");//设备名称
$request->setTimeOut(5000); //超时时间，单位毫秒.如果超过这个时间设备没反应则返回"TIMEOUT"
$request->setRpcContent("aGVsbG8gd29ybGQ=");//推送给设备的数据.数据要求二进制数据做一次BASE64编码.(示例里面是"helloworld"编码后的值)
$response = $client->getAcsResponse($request);
print_r("\r\n");
print_r($response);
```

- python

```
request = RevertRpcRequest.RevertRpcRequest()
request.set_accept_format('json') // 设置返回数据格式，默认为XML
request.set_ProductKey('123456')
request.set_DeviceName('xxxxxxx') // 设备名称
request.set_RpcContent('aGVsbG8gd29ybGQ=') // 推送给设备的数据.数据要求二进制数据做一次BASE64编码.(示例里面是"helloworld"编码后的值)
request.set_TimeOut(1000) // 超时时间，单位毫秒.如果超过这个时间设备没反应则返回"TIMEOUT"
result = clt.do_action(request)
print 'revert rpc : ' + result
```

返回示例

json示例

```
{
"Rpccode":"UNKONW",
"Success":true,
"ResponseContent":"bm90IGZvdW5kIHJvdXRlciByZWNvcmQu"
}
```

XML示例

```
<RevertRpcResponse>
<Rpccode>UNKONW</Rpccode>
<Success>true</Success>
<ResponseContent>bm90IGZvdW5kIHJvdXRlciByZWNvcmQu</ResponseContent>
</RevertRpcResponse>
```

设备状态回调数据格式

如果您需要订阅设备激活或者上下线状态通知，可以在控制台配置回调地址，请参考控制台->服务配置, 支持两种方式来订阅设备状态，配置设备状态Topic 或 状态回调地址。两个方式的区别：

topic方式（推荐）	回调地址方式
服务器会把设备状态数据发送到您指定的topic主题，可以通过配置规则引擎把数据推送到mns服务器队列，然后您的服务器去消费mns消息即可。消息不丢失，稳定可靠，在高峰时有削峰填谷作用	服务器把状态数据直接通过http调用您的服务器，需要您自己实现webserver暴露一个http地址，流量大时您需要考虑使用lvs负载均衡，否则大量消息可能会导致您的服务器承受巨大压力

topic方式服务器端订阅请参考服务端订阅消息

对应的回调数据格式参考:

- 设备激活数据格式

参数名为 data和sign, data为json字符串格式，如下.

```
data={
"status":"active",
"productKey":"xxx",
"deviceName":"sss",
"deviceId":"deviceId",
"time":"2015-12-23 00:00:00" //发送回调时间点.
}
```

- 设备上下线回调

参数名为 data和sign, data为json字符串格式，如下.

```
data=
{
"status":"online"(或offline),
"productKey":"xxx",
"deviceName":"sss",
"deviceId":"deviceId",
"time":"2015-12-23 00:00:00" //发送回调时间点.
}
```

如果是回调服务器地址方式，阿里云会以POST方式将上述信息发送给配置的服务器地址，请将对应地址设置为允许接收POST请求，否则会导致数据无法投递过去.

sign说明

此参数仅针对回调服务器地址方式传递，返回参数sign后，客户端可以按照一样的逻辑进行签名对比，以便识别请求来至于合法的阿里云服务端。

具体的加签逻辑如下:md5_32(productKey+data+productSecret)

举例:

```
ProductKey=A,ProductSecret=B,data=C
sign=md5_32(ACB)
```