ALIBABA CLOUD

阿里云

物联网无线连接服务

API参考

文档版本：20220630

阿里云

法律声明

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

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

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

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

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

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

6. 如若发现本文档存在任何错误，请与阿里云取得直接联系。

通用约定

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

目录

1.API概览

物联网无线连接服务提供以下相关API接口。

API	描述
QueryCardFlowInfo	调用QueryCardFlowInfo查询物联网卡的流量。
DoSendIotSms	调用DoSendIotSms发送物联网卡短信。
QueryIotCardOfferDtl	调用QueryIotCardOfferDtl查询物联网卡当前时间有效套餐的列表。
DoIotPostImei	调用DoIotPostImei提交物联网卡设备信息。
DoIotIsImeiExist	调用DoIotIsImeiExist判重物联网卡的设备信息。
QueryPersonalInfo	调用QueryPersonalInfo查询物联网卡实名认证的结果。
DoIotSetRemindConfig	调用DoIotSetRemindConfig设置物联网用量预警阈值。
DoIotUserStopResume	调用DoIotUserStopResume管理停复机。
DoIotChgBindOrUnBind	调用DoIotChgBindOrUnBind解绑或换绑物联网卡。
QueryCardInfo	调用QueryCardInfo查询物联网卡的明细信息。
DoIotRecharge	调用DoIotRecharge给物联网卡充值流量。

2.HTTP协议及签名

为方便其它语言能快速实现，在此增加了HTTP的详解

一、前言说明

云通信产品平台都提供了通过HTTP方式支撑多语言对接SDK 下面以判断设备是否存在（DoIotIsImeiExist） 举例

对应的协议是：

发送API采用Rest协议：签名算法使用了阿里云的POP协议

二、协议说明

由于POP的签名算法比较复杂，下面这里结合一判断设备是否存在（DoIotIsImeiExist）API一并详细说明

2.1判断设备是否存在（DoIotIsImeiExist） API的HTTP协议包示例

请求示例：

```
GET /?Signature=YjypUPcYBwdmb%2FLMWfrVx%2B61RKY%3D&AccessKeyId=testId&Action=DoIotIsImeiExist&Format=XML&Imei=123456&SignatureMethod=HMAC-SHA1&Signat
ureNonce=ea658de8-7f59-4eb2-923c-70e07f947e62&SignatureVersion=1.0&Timestamp=2018-07-11T08%3A17%3A08Z&Version=2017-11-11
HTTP/1.1
Host: dyiotapi.aliyuncs.com
...
```

返回示例：

```
HTTP/1.1 200 OK
...
<?xml version="1.0" encoding="utf-8"?>
<DoIotIsImeiExistResponse>
  <Message>OK</Message>
  <RequestId>E8534574-7381-4810-8F70-65B37BBA8970</RequestId>
  <Code>OK</Code>
  <IotImeiExist>
    <IsImeiExist>true</IsImeiExist>
  </IotImeiExist>
</DoIotIsImeiExistResponse>
```

2.2 判断设备是否存在（DoIotIsImeiExist）Rest请求参数

如HTTP示例包中，请求的参数可以分两大块：系统参数和业务参数。

系统参数

系统参数为POP协议的基本参数，有

参数KEY	是否必填	说明
AccessKeyId	是	访问密钥 ID。AccessKey 用于调用 API。
Timestamp	是	格式为：yyyy-MM-dd'T'HH:mm:ss'Z'；时区为：GMT
Format	否	没传默认为JSON，可选填值：XML
SignatureMethod	是	建议固定值：HMAC-SHA1
SignatureVersion	是	建议固定值：1.0
SignatureNonce	是	用于请求的防重放攻击，每次请求唯一，JAVA语言建议用：java.util.UUID.randomUUID()生成即可
Signature	是	最终生成的签名结果值

业务参数

参数KEY	是否必填	说明
Action	是	API的命名，固定值，如设备信息判重API的值为：DoIotIsImeiExist
Version	是	API的版本，固定值，如物联卡设备信息判重API的值为：2017-11-11
IMEI	是	物联网卡序列号， 例如123123

2.3 生成签名

签名是为了让请求合法，共分为五步：

第一步：请求参数

1. 请求参数包括系统参数和业务参数，不要遗漏
2. 请求参数中不允许出现以Signature为key的参数。参考代码如下

```
String accessKeyId = "testId";String accessSecret = "testSecret";
java.text.SimpleDateFormat df = new java.text.SimpleDateFormat("yyyy-MM-dd'T'HH:mm:ss'Z'");df.setTimeZone(new java.util.SimpleTimeZone(0, "GMT"));
// 这里一定要设置GMT时区
java.util.Map paras = new java.util.HashMap();
// 1. 系统参数paras.put("SignatureMethod", "HMAC-SHA1");paras.put("SignatureNonce", java.util.UUID.randomUUID().toString());paras.put("AccessKeyId", a
ccessKeyId);paras.put("SignatureVersion", "1.0");paras.put("Timestamp", df.format(new java.util.Date()));paras.put("Format", "XML");
// 2. 业务API参数paras.put("Action", "DoIotIsImeiExist");paras.put("Version", "2017-11-11"); paras.put("Imei", "123123");
// 3. 去除签名关键字Keyif (paras.containsKey("Signature")) paras.remove("Signature");
```

第二步：根据参数Key排序（顺序）

参考代码如下：

```
java.util.TreeMap sortParas = new java.util.TreeMap();sortParas.putAll(paras);
```

第三步：构造待签名的请求串

首先介绍下面会用到的特殊URL编码这个是POP特殊的一种规则，即在一般的URLEncode后再增加三种字符替换：加号（+）替换成 %20、星号（*）替换成 %2A、%7E 替换回波浪号（~）参考代码如下：

```
public static String specialUrlEncode(String value) throws Exception { return java.net.URLEncoder.encode(value, "UTF-8").replace("+",
"%20").replace("*", "%2A").replace("%7E", "~");}
```

构造待签名的请求串这里有两步动作第1步，把排序后的参数顺序拼接成如下格式：

```
specialUrlEncode(参数Key) + "=" + specialUrlEncode(参数值)
```

参考代码如下：

```
java.util.Iterator it = sortParas.keySet().iterator();StringBuilder sortQueryStringTmp = new StringBuilder();while (it.hasNext()) { String key =
it.next(); sortQueryStringTmp.append("&").append(specialUrlEncode(key)).append("=").append(specialUrlEncode(paras.get(key)));}String sortedQueryString
= sortQueryStringTmp.substring(1);// 去除第一个多余的&符号
```

打印上面的sortQueryString结果如下：

```
AccessKeyId%3DtestId&Action%3DDoIotIsImeiExist&Format%3DXML&Imei%3D123123&SignatureMethod%3DHMAC-SHA1&SignatureNonce%3De538f847-fa76-430b-a151-
ff88dd1e932e&SignatureVersion%3D1.0&Timestamp%3D2018-07-11T09%253A47%253A46Z&Version%3D2017-11-11
```

对应的未URL编码的值（方便用户对比）：

```
AccessKeyId=testId&Action=DoIotIsImeiExist&Format=XML&Imei=123123&SignatureMethod=HMAC-SHA1&SignatureNonce=e538f847-fa76-430b-a151-
ff88dd1e932e&SignatureVersion=1.0&Timestamp=2018-07-11T09%3A47%3A46Z&Version=2017-11-11
```

第2步，按POP的签名规则拼接成最终的待签名串，规则如下：

```
HTTPMethod + "&" + specialUrlEncode("/") + "&" +  specialUrlEncode(sortedQueryString)
```

参考代码如下：

```
StringBuilder stringToSign = new
StringBuilder();stringToSign.append("GET").append("&");stringToSign.append(specialUrlEncode("/")).append("&");stringToSign.append(specialUrlEncode(sorted
```

这就完成了待签名的请求字符串了

打印结果如下：

```
GET&%2F&AccessKeyId%3DtestId&Action%3DDoIotIsImeiExist&Format%3DXML&Imei%3D123123&SignatureMethod%3DHMAC-SHA1&SignatureNonce%3De538f847-fa76-430b-
a151-ff88dd1e932e&SignatureVersion%3D1.0&Timestamp%3D2018-07-11T09%253A47%253A46Z&Version%3D2017-11-11
```

第四步：签名

签名采用HmacSHA1算法 + Base64，编码采用：UTF-8参考代码如下：

```
String sign = sign(accessSecret + "&", stringToSign.toString());
```

```
public static String sign(String accessSecret, String stringToSign) throws Exception { javax.crypto.Mac mac =
javax.crypto.Mac.getInstance("HmacSHA1"); mac.init(new javax.crypto.spec.SecretKeySpec(accessSecret.getBytes("UTF-8"), "HmacSHA1")); byte[] signData =
mac.doFinal(stringToSign.getBytes("UTF-8")); return new sun.misc.BASE64Encoder().encode(signData);}
```

参数说明：

1. `accessSecret`：你的AccessKeyId对应的密钥AccessSecret，特别说明：POP要求需要后面多加一个“&”字符，即 `accessSecret + "&"`
2. `stringToSign`：即第三步生成的待签名请求串签名后的结果打印如下：`bsPn2jLTdPMtVrHIVFL9K1SiHBw=`

第五步：增加签名结果到请求参数中，发送请求

注意：签名也要做特殊URL编码

```
String Signature = specialUrlEncode(sign);// bsPn2jLTdPMtVrHIVFL9K1SiHBw%3D
```

最终完整的GET请求HTTP为：

```
http://dyiotapi.aliyuncs.com/?
Signature=bsPn2jLTdPMtVrHIVFL9K1SiHBw%3D&AccessKeyId=testId&Action=DoIotIsImeiExist&Format=XML&Imei=123123&SignatureMethod=HMAC-
SHA1&SignatureNonce=e538f847-fa76-430b-a151-ff88dd1e932e&SignatureVersion=1.0&Timestamp=2018-07-11T09%3A47%3A46Z&Version=2017-11-11
```

三、附加完整的Java签名Demo代码

```
public static void main(String[] args) throws Exception {
        String accessKeyId = "testId";
        String accessSecret = "testSecret";
        java.text.SimpleDateFormat df = new java.text.SimpleDateFormat("yyyy-MM-dd'T'HH:mm:ss'Z'");
        df.setTimeZone(new java.util.SimpleTimeZone(0, "GMT"));// 这里一定要设置GMT时区
        java.util.Map<String, String> paras = new java.util.HashMap<String, String>();
        // 1. 系统参数
        paras.put("SignatureMethod", "HMAC-SHA1");
        paras.put("SignatureNonce", java.util.UUID.randomUUID().toString());
        paras.put("AccessKeyId", accessKeyId);
        paras.put("SignatureVersion", "1.0");
        paras.put("Timestamp", df.format(new java.util.Date()));
        paras.put("Format", "XML");
        // 2. 业务API参数
        paras.put("Action", "DoIotIsImeiExist");
        paras.put("Version", "2017-11-11");

        paras.put("Imei", "123123");
        // 3. 去除签名关键字Key
        if (paras.containsKey("Signature"))
            paras.remove("Signature");
        // 4. 参数KEY排序
        java.util.TreeMap<String, String> sortParas = new java.util.TreeMap<String, String>();
        sortParas.putAll(paras);
        // 5. 构造待签名的字符串
        java.util.Iterator<String> it = sortParas.keySet().iterator();
        StringBuilder sortQueryStringTmp = new StringBuilder();
        while (it.hasNext()) {
            String key = it.next();
            sortQueryStringTmp.append("&").append(specialUrlEncode(key)).append("=").append(specialUrlEncode(paras.get(key)));
        }
        String sortedQueryString = sortQueryStringTmp.substring(1);// 去除第一个多余的&符号
        StringBuilder stringToSign = new StringBuilder();
        stringToSign.append("GET").append("&");
        stringToSign.append(specialUrlEncode("/")).append("&");
        stringToSign.append(specialUrlEncode(sortedQueryString));
        String sign = sign(accessSecret + "&", stringToSign.toString());
        // 6. 签名最后也要做特殊URL编码
        String signature = specialUrlEncode(sign);
        System.out.println(paras.get("SignatureNonce"));
        System.out.println("\r\n=========\r\n");
        System.out.println(paras.get("Timestamp"));
        System.out.println("\r\n=========\r\n");
        System.out.println(sortedQueryString);
        System.out.println("\r\n=========\r\n");
        System.out.println(stringToSign.toString());
        System.out.println("\r\n=========\r\n");
        System.out.println(sign);
        System.out.println("\r\n=========\r\n");
        System.out.println(signature);
        System.out.println("\r\n=========\r\n");
        // 最终打印出合法GET请求的URL
        System.out.println("http://dyiotapi.aliyuncs.com/?Signature=" + signature + sortQueryStringTmp);
    }
    public static String specialUrlEncode(String value) throws Exception {
        return java.net.URLEncoder.encode(value, "UTF-8").replace("+", "%20").replace("*", "%2A").replace("%7E", "~");
    }
    public static String sign(String accessSecret, String stringToSign) throws Exception {
        javax.crypto.Mac mac = javax.crypto.Mac.getInstance("HmacSHA1");
        mac.init(new javax.crypto.spec.SecretKeySpec(accessSecret.getBytes("UTF-8"), "HmacSHA1"));
        byte[] signData = mac.doFinal(stringToSign.getBytes("UTF-8"));
        return new sun.misc.BASE64Encoder().encode(signData);
    }
```

3.公共请求及返回参数

本文介绍物联网无线连接服务云端API的公共请求参数和公共返回参数。

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`，本版本对应为`2017-11-11`
AccessKeyId	String	是	阿里云颁发给用户的访问服务所用的密钥ID
Signature	String	是	签名结果串，关于签名的计算方法，请参见签名机制
SignatureMethod	String	是	签名方式，目前支持HMAC-SHA1
Timestamp	String	是	请求的时间戳。日期格式按照[ISO8601]标准表示，并需要使用UTC时间。格式为`YYYY-MM-DDThh:mm:ssZ`例如，`2018-11-23T04:00:00Z`（为北京时间2018年11月23日12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	否	机房信息

示例

```
https://dyiotapi.aliyuncs.com/
?Format=XML
&Version=2017-11-11
&Signature=Pc5WB8gokVn0xfeu%2FZV%2BiNM1dgI%3D
&SignatureMethod=HMAC-SHA1
&SignatureNonce=e1b44502-6d13-4433-9493-69eeb068e955
&SignatureVersion=1.0
&AccessKeyId=key-test
&Timestamp=2018-11-23T12:00:00Z
```

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 */
}
```

4.云端Java SDK使用说明

物联网无线连接服务提供Java SDK，方便开发者使用Java程序操作物联网无线连接服务。开发者可以使用Maven依赖添加SDK。

1. 安装 Java 开发环境。您可以从Java 官方网站下载，并按说明安装Java开发环境。
2. 安装物联网络管理平台Java SDK。

- 访问 Apache Maven 官网下载 Maven 软件。
- 添加 Maven 项目依赖。物联网无线连接服务控制台SDK的Maven依赖坐标：

```
<dependency>
   <groupId>com.aliyun</groupId>
   <artifactId>aliyun-java-sdk-dyiotapi</artifactId>
   <version>2.1.1</version>
</dependency>
```

阿里云云端公共SDK的Maven依赖坐标：

```
<dependency>
    <groupId>com.aliyun</groupId>
    <artifactId>aliyun-java-sdk-core</artifactId>
    <version>4.1.1</version>
</dependency>
```

初始化 SDK

1. 为了使用物联卡 SDK，您必须申请阿里云的访问密钥。
2. 阿里云访问密钥是阿里云为用户使用 API（非控制台）来访问其云资源设计的“安全口令”。您可以用它来签名 API 请求内容以通过服务端的安全验证。
3. 该访问密钥成对（AccessKeyId 与 AccessKeySecret）生成和使用。每个阿里云用户可以创建多对访问密钥，且可随时启用（Active）、禁用（Inactive）或者删除已经生成的访问密钥对。
4. 您可以通过阿里云控制台的 密钥管理页面 创建、管理所有的访问密钥对，且保证它处于“启用”状态。由于访问密钥是阿里云对 API 请求进行安全验证的关键因子，请妥善保管你的访问密钥。如果某些密钥对出现泄漏风险，建议及时删除该密钥对并生成新的替代密钥对。

注：有备注无需修改的位置请勿改动。

```
//设置超时时间-可自行调整
System.setProperty("sun.net.client.defaultConnectTimeout", "10000");
System.setProperty("sun.net.client.defaultReadTimeout", "10000");
//初始化ascClient需要的几个参数
final String product = "Dyiotapi";//物联卡API产品名称（短信产品名固定，无需修改）
final String domain = "dyiotapi.aliyuncs.com";//物联卡API产品域名（接口地址固定，无需修改）
//替换成你的AK
final String accessKeyId = "yourAccessKeyId";//你的accessKeyId,参考本文档步骤2
final String accessKeySecret = "yourAccessKeySecret";//你的accessKeySecret，参考本文档步骤2
//初始化ascClient,暂时不支持多region（请勿修改）
IClientProfile profile = DefaultProfile.getProfile("cn-hangzhou", accessKeyId,
accessKeySecret);
DefaultProfile.addEndpoint("cn-hangzhou", "cn-hangzhou", product, domain);
IAcsClient acsClient = new DefaultAcsClient(profile);
```

下文以调用 `QueryCardHistoryFlowInfo API` 方法查询物联网卡历史流量数据为例，描述调用API的方法。

```
//构建该API特定的请求参数
 QueryCardHistoryFlowInfoRequest request = new QueryCardHistoryFlowInfoRequest();
//填入你要查询的iccid值
 request.setIccid("89860403101801032902");
 //填入查询开始时间
 request.setStartTime("202001");
 //填入查询结束时间
 request.setEndTime("202006");
 try {
 QueryCardHistoryFlowInfoResponse  response = acsClient.getAcsResponse(request);
        } catch (ClientException e) {
            e.printStackTrace();
        }
if(response.getSuccess()) {
 System.out.println(response);
}
```

5.QueryCardFlowInfo

调用QueryCardFlowInfo查询物联网卡的流量。

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
Iccid	String	是	89860617030017300390	要查询的物联卡对应的Iccid编码。
AccessKeyId	String	否	your_accesskey_id	您的AccessKey ID。
Action	String	否	QueryCardFlowInfo	系统规定参数。取值：**QueryCardFlowInfo**。

返回数据

名称	类型	示例值	描述
CardFlowInfos			物联网卡流量的信息列表。
CardFlowInfo			物联网卡流量的信息列表。
ExpireDate	String	20180504235959	资源失效日期。
FlowResource	Long	1048576	资源总量，流量单位为KB。
FlowUsed	Long	0	资源使用量，流量单位为KB。
ResName	String	物联网-联通-Internet-自定义流量包	资源名称。
ResourceType	String	6700001	资源类型编码。6700001代表流量。
RestOfFlow	Long	1048576	资源剩余量，流量单位为KB。
SmsUsed	Long	389	短信使用量。以条为单位。
ValidDate	String	20171106174912	资源生效日期。
VoiceTotal	Long	60	语音套餐总量，以分钟为单位。
VoiceUsed	Long	25	语音使用量，以分钟为单位。
Code	String	OK	状态码。
Message	String	ServiceUnavailable	状态码的描述。
RequestId	String	1C63F16F-D376-4065-816E-3E56CDD13FEB	请求ID。

示例

请求示例

```
http(s)://[Endpoint]/?Action=QueryCardFlowInfo
&Iccid=89860617030017300390
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<QueryCardFlowInfoResponse>
   <RequestId>1C63F16F-D376-4065-816E-3E56CDD13FEB</RequestId>
   <Code>OK</Code>
   <Message>ServiceUnavailable</Message>
   <CardFlowInfos>
      <ResourceType>6700001</ResourceType>
      <ResName>物联网-联通-Internet-自定义流量包</ResName>
      <FlowResource>1048576</FlowResource>
      <RestOfFlow>1048576</RestOfFlow>
      <FlowUsed>0</FlowUsed>
      <ValidDate>20171106174912</ValidDate>
      <ExpireDate>20180504235959</ExpireDate>
      <SmsUsed>389</SmsUsed>
      <VoiceUsed>25</VoiceUsed>
      <VoiceTotal>60</VoiceTotal>
   </CardFlowInfos>
</QueryCardFlowInfoResponse>
```

`JSON` 格式

```
{
 "Message":"ServiceUnavailable",
 "CardFlowInfos":[
  {
   "VoiceTotal":60,
   "ExpireDate":"20180504235959",
   "ResName":"物联网-联通-Internet-自定义流量包",
   "ResourceType":"6700001",
   "FlowUsed":0,
   "RestOfFlow":1048576,
   "SmsUsed":389,
   "ValidDate":"20171106174912",
   "VoiceUsed":25,
   "FlowResource":1048576
  }
 ],
 "RequestId":"1C63F16F-D376-4065-816E-3E56CDD13FEB",
 "Code":"OK"
}
```

错误码

访问错误中心查看更多错误码。

6.DoSendIotSms

调用DoSendIotSms发送物联网卡短信。

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	DoSendIotSms	系统规定参数。取值：**DoSendIotSms**。
SignName	String	是	your_sign_name	短信签名。
TemplateCode	String	是	your_template_code	短信对应的模板code。 数据短信和普通短信依据的模板code不同。
PhoneNumbers	String	是	1064633113571	接受短信的物联卡MSISDN。
TemplateParam	String	是	{"key":"value"}	渲染短信内容的模板参数，该参数是一个JSON字符串，且key只能是content。

返回数据

名称	类型	示例值	描述
Code	String	OK	状态码。
Module	String	101510614255450461^0	发送短信成功后，返回的全局唯一序列。
Message	String	请求成功	状态码的描述。
RequestId	String	F0EB4AA8-3284-41EB-BECE-17D12088EA0A	请求ID。

示例

请求示例

```
http(s)://[Endpoint]/?Action=DoSendIotSms
&PhoneNumbers=1064633113571
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
HTTP/1.1 200 OK
Content-Type:application/xml
<?xml version="1.0" encoding="UTF-8" ?>
<DoSendIotSmsResponse>
 <RequestId>F0EB4AA8-3284-41EB-BECE-17D12088EA0A</RequestId>
 <Code>OK</Code>
 <Message>OK</Message>
 <Module>101510614255450461^0</Module>
</DoSendIotSmsResponse>
```

`JSON` 格式

```
HTTP/1.1 200 OK
Content-Type:application/json
{
  "RequestId" : "F0EB4AA8-3284-41EB-BECE-17D12088EA0A",
  "Code" : "OK",
  "Message" : "OK",
  "Module" : "101510614255450461^0"
}
```

7.QueryIotCardOfferDtl

调用QueryIotCardOfferDtl查询物联网卡当前时间有效套餐的列表。

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
Iccid	String	是	89860617030017300390	要查询的物联网卡对应的Iccid编码。
AccessKeyId	String	否	your_accesskey_id	您的AccessKey ID。
Action	String	否	QueryIotCardOfferDtl	系统规定参数。取值：**QueryIotCardOfferDtl**。

返回数据

名称	类型	示例值	描述
CardOfferDetail			有效套餐列表。
detail			有效套餐列表。
EffectiveTime	String	2017-11-06 17:49:12	生效时间
ExpireTime	String	2099-12-31 23:59:59	失效时间
OfferId	String	12312312	套餐编号
OfferName	String	XXX套餐	套餐名称
OrderTime	String	2017-11-06 17:47:12	订购时间
Code	String	OK	状态码。
Message	String	OK	状态码的描述。
RequestId	String	A9D2A751-8207-4951-A138-A2D5B0208728	请求ID。

示例

请求示例

```
http(s)://[Endpoint]/?Action=QueryIotCardOfferDtl
&Iccid=89860617030017300390
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<QueryIotCardOfferDtlResponse>
  <RequestId>A9D2A751-8207-4951-A138-A2D5B0208728</RequestId>
   <Code>OK</Code>
   <Message>OK</Message>
   <CardOfferDetail>
      <detail>
         <ExpireTime>2099-12-31 23:59:59</ExpireTime>
         <EffectiveTime>2017-11-06 17:49:12</EffectiveTime>
         <OrderTime>2017-11-06 17:47:12</OrderTime>
         <OfferName>物联网-联通(internet)-基础套餐</OfferName>
         <OfferId>22010000142006</OfferId>
      </detail>
      <detail>
         <ExpireTime>2018-05-04 23:59:59</ExpireTime>
         <EffectiveTime>2017-11-06 17:49:12</EffectiveTime>
         <OrderTime>2017-11-06 17:47:12</OrderTime>
         <OfferName>物联网-联通(internet)流量180天-1G</OfferName>
         <OfferId>22010000219106</OfferId>
      </detail>
   </CardOfferDetail>
</QueryIotCardOfferDtlResponse>
```

`JSON` 格式

```
{
 "CardOfferDetail":{
  "detail":[
   {
    "OrderTime":"2017-11-06 17:47:12",
    "OfferId":"22010000142006",
    "ExpireTime":"2099-12-31 23:59:59",
    "OfferName":"物联网-联通(internet)-基础套餐",
    "EffectiveTime":"2017-11-06 17:49:12"
   },
   {
    "OrderTime":"2017-11-06 17:47:12",
    "OfferId":"22010000219106",
    "ExpireTime":"2018-05-04 23:59:59",
    "OfferName":"物联网-联通(internet)流量180天-1G",
    "EffectiveTime":"2017-11-06 17:49:12"
   }
  ]
 },
 "Message":"OK",
 "RequestId":"A9D2A751-8207-4951-A138-A2D5B0208728",
 "Code":"OK"
}
```

错误码

访问错误中心查看更多错误码。

8.DoIotPostImei

调用DoIotPostImei提交物联网卡设备信息。

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
Imei	String	是	861234567890	要上传设备的IMEI编码。
AccessKeyId	String	否	your_accesskey_id	您的AccessKey ID。
Action	String	否	DoIotPostImei	系统规定参数。取值：**DoIotPostImei**。
Comments	String	否	xxx	备注。
DeviceType	String	否	xxx	设备类型。

返回数据

名称	类型	示例值	描述
Code	String	OK	状态码。
IotPostImei			该对象为一个Object。
IsPostSuccess	Boolean	true	提交是否成功。
Message	String	ServiceUnavailable	状态码的描述。
RequestId	String	1C63F16F-D376-4065-816E-3E56CDD13FEB	请求ID。

示例

请求示例

```
http(s)://[Endpoint]/?Action=DoIotPostImei
&Imei=861234567890
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<DoIotPostImeiResponse>
   <RequestId>1C63F16F-D376-4065-816E-3E56CDD13FEB</RequestId>
   <Code>OK</Code>
   <Message>ServiceUnavailable</Message>
   <IotPostImei>
      <IsPostSuccess>true</IsPostSuccess>
   </IotPostImei>
</DoIotPostImeiResponse>
```

`JSON` 格式

```
{
 "Message":"ServiceUnavailable",
 "RequestId":"1C63F16F-D376-4065-816E-3E56CDD13FEB",
 "IotPostImei":{
  "IsPostSuccess":true
 },
 "Code":"OK"
}
```

错误码

访问错误中心查看更多错误码。

9.DoIotIsImeiExist

调用DoIotIsImeiExist判重物联网卡的设备信息。

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
Imei	String	是	861234567890	要判重设备的IMEI。
AccessKeyId	String	否	your_accesskey_id	您的AccessKey ID。
Action	String	否	DoIotIsImeiExist	系统规定参数。取值：**DoIotIsImeiExist**。

返回数据

名称	类型	示例值	描述
Code	String	OK	状态码。
IotImeiExist			该对象为一个Object。
IsImeiExist	Boolean	true	是否存在。**true**表示存在，**false**表示不存在。
Message	String	ServiceUnavailable	状态码的描述。
RequestId	String	1C63F16F-D376-4065-816E-3E56CDD13FEB	请求ID。

示例

请求示例

```
http(s)://[Endpoint]/?Action=DoIotIsImeiExist
&Imei=861234567890
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<DoIotIsImeiExistResponse>
   <RequestId>1C63F16F-D376-4065-816E-3E56CDD13FEB</RequestId>
   <Code>OK</Code>
   <Message>ServiceUnavailable</Message>
   <IotImeiExist>
      <IsImeiExist>true</IsImeiExist>
   </IotImeiExist>
</DoIotIsImeiExistResponse>
```

`JSON` 格式

```
{
 "Message":"ServiceUnavailable",
 "RequestId":"1C63F16F-D376-4065-816E-3E56CDD13FEB",
 "Code":"OK",
 "IotImeiExist":{
  "IsImeiExist":true
 }
}
```

错误码

访问错误中心查看更多错误码。

10.QueryPersonalInfo

调用QueryPersonalInfo查询物联网卡实名认证的结果。

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
Iccid	String	是	89860617030017300390	要充值的物联卡对应的Iccid编码。
AccessKeyId	String	否	your_accesskey_id	您的AccessKey ID。
Action	String	否	QueryPersonalInfo	系统规定参数。取值：**QueryPersonalInfo**。

返回数据

名称	类型	示例值	描述
Code	String	OK	状态码。
Message	String	ServiceUnavailable	状态码的描述。
Module	String	{\"cardName\":\"***\",\"cardNum\":\"***\",\"verifyStatus\":\"3\",\"cardType\":\"1\",\"authTime\":\"2017-12-12 00:00:00\"}	认证详情。
RequestId	String	475C4B95-FB6E-4B2A-B0C6-8AAA136B323A	请求ID。

示例

请求示例

```
http(s)://[Endpoint]/?Action=QueryPersonalInfo
&Iccid=89860617030017300390
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<QueryPersonalInfoResponse>
    <RequestId>475C4B95-FB6E-4B2A-B0C6-8AAA136B323A</RequestId>
    <Code>OK</Code>
    <Message>ServiceUnavailable</Message>
    <Module>{"cardName":"***","cardNum":"***","verifyStatus":"3","cardType":"1","authTime":"2017-12-12 00:00:00"}</Module>
</QueryPersonalInfoResponse>
```

`JSON` 格式

```
{
 "Message":"ServiceUnavailable",
 "RequestId":"475C4B95-FB6E-4B2A-B0C6-8AAA136B323A",
 "Module":"{\"cardName\":\"***\",\"cardNum\":\"***\",\"verifyStatus\":\"3\",\"cardType\":\"1\",\"authTime\":\"2017-12-12 00:00:00\"}",
 "Code":"OK"
}
```

错误码

访问错误中心查看更多错误码。

11.DoIotSetRemindConfig

调用DoIotSetRemindConfig设置物联网用量预警阈值。

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	DoIotSetRemindConfig	系统规定参数。取值：**DoIotSetRemindConfig**。
BizType	String	是	GROUP	业务类型。取值： • **ICCID**：指单卡预警。 • **GROUP**：指分组预警。 • **POOL**：指流量池预警。
BizId	String	是	GROUP_ID	根据不同的biztype选择对应的bizId。 • ICCID：指对应的物联网卡号ICCID。 • GROUP：指对应的物联网卡分组编号GROUP_ID。（请联系您的业务对接人处理线下获取） • POOL：指对应的流量池编号，一般为流量池对应的计费号。（请联系您的业务对接人处理线下获取）
OperationType	String	是	1	操作类型。 1-新增，2-更新，3-删除
ConfigInfo	String	否	50,70,90	需要提醒的阈值，会有多个，如达到50%，70%，90%时提醒，则设置为50,70,90，当OperaterType为1、2时必填。

返回数据

名称	类型	示例值	描述
Code	String	200	返回结果编码。
Message	String	是	返回消息。
RequestId	String	2430E47F-46A2-48C9-9587-E17C56FBCFE0	请求ID。

示例

请求示例

```
http(s)://[Endpoint]/?Action=DoIotSetRemindConfig
&BizId=GROUP_ID
&BizType=GROUP
&OperationType=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
<DoIotSetRemindConfigResponse>
 <RequestId>475C4B95-FB6E-4B2A-B0C6-8AAA136B323A</RequestId>
 <Code>OK</Code>
 <Message>ServiceUnavailable</Message>
</DoIotSetRemindConfigResponse>
```

`JSON` **格式**

```
HTTP/1.1 200 OK
Content-Type:application/json
{
  "RequestId" : "475C4B95-FB6E-4B2A-B0C6-8AAA136B323A",
  "Code" : "OK",
  "Message" : "ServiceUnavailable"
}
```

12.DoIotSetAbsoluteRemindConfig

调用DoIotSetAbsoluteRemindConfig设置物联网用量预警绝对值阈值。

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	DoIotSetAbsoluteRemindConfig	系统规定参数。取值：**DoIotSetAbsoluteRemindConfig**。
BizType	String	是	GROUP	业务类型： • ICCID 设置单卡预警 • GROUP 设置卡分组预警 • POOLCARD 设置流量池中所有卡的预警
BizId	String	是	GROUP_ID	根据不同的biztype选择对应的bizId， • ICCID指卡ICCID • GROUP指物联网卡分组编号GROUP_ID（请联系您的业务对接人处理线下获取） • POOLCARD，流量池编号，一般为流量池对应的计费号（请联系您的业务对接人处理线下获取）
ConfigInfo	String	否	50,100,200	需要设置的预警阈值，单位M，最多支持设置10档； 比如：50M，100M，200M时需要提醒，则传值50,100,200；

返回数据

名称	类型	示例值	描述
Code	String	200	返回结果编码
Message	String	是	返回消息
RequestId	String	2430E47F-46A2-48C9-9587-E17C56FBCFE0	请求ID

示例

请求示例

```
http(s)://[Endpoint]/?Action=DoIotSetAbsoluteRemindConfig
&BizId=GROUP_ID
&BizType=GROUP
&ConfigInfo=50,100,200
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
HTTP/1.1 200 OK
Content-Type:application/xml
<DoIotSetAbsoluteRemindConfigResponse>
 <RequestId>475C4B95-FB6E-4B2A-B0C6-8AAA136B323A</RequestId>
 <Code>OK</Code>
 <Message>ServiceUnavailable</Message>
</DoIotSetAbsoluteRemindConfigResponse>
```

`JSON` 格式

```
HTTP/1.1 200 OK
Content-Type:application/json
{
  "RequestId" : "475C4B95-FB6E-4B2A-B0C6-8AAA136B323A",
  "Code" : "OK",
  "Message" : "ServiceUnavailable"
}
```

13.DoIotUserStopResume

调用DoIotUserStopResume管理停复机。

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
Iccid	String	是	89860617030017300390	要查询的物联卡对应的Iccid编码。
OptionType	String	是	MANAGE_STOP	• **MANAGE_STOP**：管理停 • **MANAGE_RESUME**：管理复
AccessKeyId	String	否	your_accesskey_id	您的AccessKey ID。
Action	String	否	DoIotUserStopResume	系统规定参数。取值：**DoIotUserStopResume**。

返回数据

名称	类型	示例值	描述
Code	String	OK	状态码。
Message	String	请求成功	状态码的描述。
RequestId	String	475C4B95-FB6E-4B2A-B0C6-8AAA136B323A	请求ID。
Result			停复操作结果。
ControlResult	Boolean	true	停复操作是否成功。

示例

请求示例

```
http(s)://[Endpoint]/?Action=DoIotUserStopResume
&Iccid=89860617030017300390
&OptionType=MANAGE_STOP
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<DoIotUserStopResumeResponse>
   <RequestId>475C4B95-FB6E-4B2A-B0C6-8AAA136B323A</RequestId>
   <Code>OK</Code>
   <Message>ServiceUnavailable</Message>
   <Result>
      <ControlResult>true</ControlResult>
   </Result>
</DoIotUserStopResumeResponse>
```

`JSON` 格式

```
{
 "Result":{
  "ControlResult":true
 },
 "Message":"ServiceUnavailable",
 "RequestId":"475C4B95-FB6E-4B2A-B0C6-8AAA136B323A",
 "Code":"OK"
}
```

错误码

访问错误中心查看更多错误码。

14.DoIotChgBindOrUnBind

调用DoIotChgBindOrUnBind解绑或换绑物联网卡。

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
Iccid	String	是	89860617030017300390	需要解绑或换绑的物联卡对应的Iccid编码。
Imei	String	是	861234567890	设备IMEI。
OpionType	String	是	chgBind	操作类型。取值： • **chgBind**：换绑 • **unBind**：解绑
AccessKeyId	String	否	your_accesskey_id	您的AccessKey ID。
Action	String	否	DoIotChgBindOrUnBind	要执行的操作，取值：**DoIotChgBindOrUnBind**。
MidChannelId	String	否	NULL	不填
NewImei	String	否	861234567890	在换绑的时候，新设备IMEI为必填。

返回数据

名称	类型	示例值	描述
Code	String	OK	状态码。 返回OK代表请求成功，其他错误码详见错误码列表。
IotModBind			返回数据
IsModSuccess	Boolean	true	操作是否成功。
Message	String	ServiceUnavailable	状态码的描述。
RequestId	String	475C4B95-FB6E-4B2A-B0C6-8AAA136B323A	请求ID。

示例

请求示例

```
http(s)://[Endpoint]/?Action=DoIotChgBindOrUnBind
&Iccid=89860617030017300390
&Imei=861234567890
&OpionType=chgBind
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<RequestId>475C4B95-FB6E-4B2A-B0C6-8AAA136B323A</RequestId>
<Code>OK</Code>
<Message>ServiceUnavailable</Message>
<IotModBind>
    <IsModSuccess>true</IsModSuccess>
</IotModBind>
```

`JSON` 格式

```
{
 "IotModBind":{
  "IsModSuccess":true
 },
 "Message":"ServiceUnavailable",
 "RequestId":"475C4B95-FB6E-4B2A-B0C6-8AAA136B323A",
 "Code":"OK"
}
```


错误码

访问错误中心查看更多错误码。

15.QueryCardInfo

调用QueryCardInfo查询物联网卡的明细信息。

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
Iccid	String	是	89860617030017300390	要查询的物联卡对应的Iccid编码。
AccessKeyId	String	否	your_accesskey_id	您的AccessKey ID。
Action	String	否	QueryCardInfo	系统规定参数。取值：**QueryCardInfo**。

返回数据

名称	类型	示例值	描述
CardInfo			物联卡的明细信息。
FirstActiveTime	String	20171106174912	激活时间。
GprsStatus	String	开	GPRS开关状态。
Iccid	String	89860617030017300390	ICCID编号。
Imsi	String	460001234567890	IMSI，国际移动用户识别码
Msisdn	String	1064633113571	MSISDN编码。
OpenTime	String	20171011000000	开户时间。
SmsStatus	String	开	短信功能开关状态（暂时未开放，结果列表无此字段）。
VoiceStatus	String	NULL	语音功能开关状态（暂时未开放，结果列表无此字段）。
Code	String	OK	状态码。
Message	String	请求成功	状态码的描述。
RequestId	String	7E31A459-E9B1-4A94-8EA0-A4C0E42BAF11	请求ID。

示例

请求示例

```
http(s)://[Endpoint]/?Action=QueryCardInfo
&Iccid=89860617030017300390
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<QueryCardInfoResponse>
   <RequestId>7E31A459-E9B1-4A94-8EA0-A4C0E42BAF11</RequestId>
   <Code>OK</Code>
   <Message>ServiceUnavailable</Message>
   <CardInfo>
      <Iccid>89860617030017300390</Iccid>
      <OpenTime>20171011000000</OpenTime>
      <FirstActiveTime>20171106174912</FirstActiveTime>
      <Msisdn>1064633113571</Msisdn>
      <GprsStatus>开</GprsStatus>
      <VoiceStatus>开</VoiceStatus>
      <SmsStatus>开</SmsStatus>
   </CardInfo>
</QueryCardInfoResponse>
```

`JSON` 格式

```
{
 "Message":"ServiceUnavailable",
 "RequestId":"7E31A459-E9B1-4A94-8EA0-A4C0E42BAF11",
 "Code":"OK",
 "CardInfo":{
  "OpenTime":"20171011000000",
  "SmsStatus":"开",
  "GprsStatus":"开",
  "VoiceStatus":"开",
  "Iccid":"89860617030017300390",
  "Msisdn":"1064633113571",
  "FirstActiveTime":"20171106174912"
 }
}
```

错误码

访问错误中心查看更多错误码。

16.DoIotRecharge

调用DoIotRecharge给物联网卡充值流量。

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
EffCode	String	是	1000	套餐生效类型。取值： • **AUTO_ORD**：自动在当前套餐失效后，续订本次套餐。若有多个未生效套餐，则本次续订是从最后一个未生效套餐开始往后续。 • **1000**：立即生效。
Iccid	String	是	89860617030017300390	要充值的物联卡对应的Iccid编码。
OfferIds	String	是	123456	充值的套餐编号。
OutId	String	是	123456	外部流水编码。
AccessKeyId	String	否	your_accesskey_id	您的AccessKey ID。
Action	String	否	DoIotRecharge	系统规定参数。取值：**DoIotRecharge**。
Amount	Long	否	0	自定义流量套餐流量数，单位M。 该参数配合自定义档位套餐使用，目前未开放填0。
OrderNum	Integer	否	12	订购份数

返回数据

名称	类型	示例值	描述
Code	String	OK	状态码。
IotRecharge			充值结果。
DoneCode	String	1093ys95199815616-1000000083ds46009-792893	处理流水。
OrderNo	String	P0000000182024044	充值订单号。
OrderResult	String	SUCCESS	结果描述。
Message	String	ServiceUnavailable	状态码的描述。
RequestId	String	475C4B95-FB6E-4B2A-B0C6-8AAA136B323A	请求ID。

示例

请求示例

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

正常返回示例

`XML` 格式

```
<DoIotRechargeResponse>
   <RequestId>475C4B95-FB6E-4B2A-B0C6-8AAA136B323A</RequestId>
   <Code>OK</Code>
   <Message>ServiceUnavailable</Message>
   <IotRecharge>
      <OrderNo>P0000000182024044</OrderNo>
      <DoneCode>1093ys95199815616-1000000083ds46009-792893</DoneCode>
      <OrderResult>SUCCESS</OrderResult>
   </IotRecharge>
</DoIotRechargeResponse>
```

`JSON` 格式

```
{
 "IotRecharge":{
  "OrderResult":"SUCCESS",
  "OrderNo":"P0000000182024044",
  "DoneCode":"1093ys95199815616-1000000083ds46009-792893"
 },
 "Message":"ServiceUnavailable",
 "RequestId":"475C4B95-FB6E-4B2A-B0C6-8AAA136B323A",
 "Code":"OK"
}
```

错误码

访问错误中心查看更多错误码。

17.DoIotUnbindResume

解绑复机。

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
Iccid	String	是	11111111111606808992	需要解绑复机的iccid。

返回数据

名称	类型	示例值	描述
Code	String	OK	状态码。
Data	Boolean	true	解绑复机的结果。
Message	String	OK	状态码的描述。
RequestId	String	5E069040-245B-4958-80CB-1B1CD22A1F98	请求id。

示例

请求示例

```
http(s)://[Endpoint]/?Action=DoIotUnbindResume
&Iccid=11111111111606808992
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<Message>OK</Message>
<RequestId>5E069040-245B-4958-80CB-1B1CD22A1F98</RequestId>
<Data>true</Data>
<Code>OK</Code>
```

`JSON` 格式

```
{"Message":"OK","RequestId":"5E069040-245B-4958-80CB-1B1CD22A1F98","Data":"true","Code":"OK"}
```

18.QueryCardsInfo

查询物联网卡信息。

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
Iccid	String	是	11111111111606808992	需要查询物联网卡信息的iccid。

返回数据

名称	类型	示例值	描述
CardsInfo	Array of CardsInfo		物联网卡信息。
FirstActiveTime	String	20201201000000	首次激活时间。
GprsStatus	String	开	gprs功能状态。
Iccid	String	11111111111606808992	物联网卡iccid。
Imsi	String	111111606808992	物联网卡imsi。
Msisdn	String	1111606808992	物联网卡msisdn。
OpenTime	String	20201201000000	开户时间。
SmsStatus	String	关	短信功能状态。
VoiceStatus	String	关	语音功能状态。
Code	String	OK	状态码。
Message	String	OK	状态码的描述。
RequestId	String	2A6D9296-7096-4DC0-B372-09D3A575A3A4	请求id。

示例

请求示例

```
http(s)://[Endpoint]/?Action=QueryCardsInfo
&Iccid=11111111111606808992
&<公共请求参数>
```

正常返回示例

`XML` 格式

```
<RequestId>3DAEFA57-3F95-414D-B153-4C521ADFD675</RequestId>
<Message>OK</Message>
<CardsInfo>
    <OpenTime>20201201000000</OpenTime>
    <Msisdn>1111606808992</Msisdn>
    <Iccid>11111111111606808992</Iccid>
    <GprsStatus>开</GprsStatus>
</CardsInfo>
<Code>OK</Code>
```

`JSON` 格式

```
{"RequestId":"3DAEFA57-3F95-414D-B153-4C521ADFD675","Message":"OK","CardsInfo":[{"OpenTime":"20201201000000","Msisdn":"1111606808992","Iccid":"11111111111606808992","GprsStatus":"开"}],"Code":"OK"}
```