



金卡智能集团股份有限公司

# 金卡智能物联网系统对接 接口文档

文件状态： <input type="checkbox"/> 草稿 <input checked="" type="checkbox"/> 正式发布 <input type="checkbox"/> 正在修改	当前版本：	2.1
	作者：	徐伟
	完成日期：	2020-02-28

金卡智能集团股份有限公司版权所有

## 版本历史

版本/状态	作者	参与者	起止日期	备注
1.7.1 历史	罗千	罗千	2018-09-25	增加价格设置接口、价格修改接口、余额同步接口
1.8 当前	罗千	罗千、包利剑	2018-10-19	1、修改返回值为响应吗+响应消息 2、注册接口增加付费模式和信用额度 3、抄表上报的补抄数据定义为标况数据，增加流量计的数据 4、抄表历史接口，增加工况数据 5、指令下发接口去掉设置上传周期和 ip 的指令 6、指令下发回调去掉指令码 7、气量金额修改为保留 4 位小数 8、调整接口返回值的一致性 9、价格设置接口增加价格周期、周期开始时间字段
1.9	赵云	赵云	2019-05-14	1、表具注册接口添加字段：地址、电话、开户日期、用户类别 2、新增表具修改接口 3、新增用户信息修改接口 4、新增对账接口
2.0	赵云	赵云	2019-05-17	新增批量获取抄表数据
2.0.1	徐伟	徐伟	2019-12-17	增加了 demo 示例，增加描述，修改了部分错误

## 目录

1.1 文档目的.....	5
1.2 文档范围.....	5
1.3 术语定义.....	5
<b>2. 接口说明.....</b>	<b>5</b>
2.1 通讯方式.....	5
2.2 接口安全.....	6
2.3 请求方式.....	6
2.4 消息交换说明.....	6
2.4.1 表具注册.....	6
2.4.2 表具注销.....	7
2.4.3 指令下发.....	8
2.4.4 指令取消.....	9
2.4.5 价格设置(价格新增和价格调价).....	9
2.4.6 价格修改(换价格).....	10
2.4.7 表具更换.....	11
2.4.8 批量获取抄表数据(批量获取).....	12
2.4.9 表具详细信息查询.....	13
2.4.10 获取日用量数据(每日结算记录).....	15
2.4.11 表具抄表上报.....	16
2.4.12 指令下发回调.....	17
2.4.13 表具告警上报.....	18
<b>3 参考附录.....</b>	<b>20</b>
附录一指令参数代码说明.....	20
附录二告警信息类型说明.....	21
附录三用户类型说明.....	21
附录四 错误信息说明.....	21
附录五 表具类型说明.....	22
<b>4 接口请求示例.....</b>	<b>23</b>
4.1 注册请求示例.....	23
4.2 注销请求示例.....	24
4.3 指令下发示例.....	25
4.4 指令取消示例.....	28
4.5 价格设置示例.....	29
4.6 价格修改示例.....	32
4.7 表具更换示例.....	33
4.8 批量获取抄表数据示例.....	34
4.9 表具详细信息查询示例.....	36
4.10 获取日用气量数据示例.....	37
4.11 抄表上报示例.....	39
4.12 指令下发回调示例.....	41

---

4.13 告警上报示例.....	42
5 安全加密算法.....	43

金卡智能版权所有

## 1. 文档介绍

### 1.1 文档目的

由于市场上物联网远传表种类繁多，通讯协议各不相同。系统中接入新的协议功能参差不齐，为了使当前的业务系统快速接入其他厂商的表，整理了此标准的接口文档，通过系统接口的方式接入新的厂家的物联网远传表具。

### 1.2 文档范围

此文档适用于物联网系统和表具厂商、产品经理、管理人员、开发人员和测试人员。

### 1.3 术语定义

CIS: 客服业务系统

DAS(DATA ACQUISITION SYSTEMS): 数据采集系统

## 2. 接口说明

### 2.1 通讯方式

- 1) 处理业务采用 HTTP 协议，HTTP 协议以 Restful 方式进行交互
- 2) 本协议字符集默认为 UTF-8，请勿使用其他字符集
- 3) 统一 HTTP 头信息

属性	类型	约束	说明
Accept	String	必选	客户端响应接收数据格式: application/json
Content-Type	String	必选	类型: application/json;charset=utf-8
AccessToken	String	必选	访问令牌（请求 api 必须，否则不必须） 使用 MD5 加密（appCode + appSecret + 时间戳） 时间戳是当前系统时间，格式"yyyyMMddHHmmss"
Authorization	String	必选	验证信息（请求 api 必须，否则不必须） 使用 Base64 编码（appCode + 冒号(英文) + 时间戳） 时间戳是当前系统时间，格式"yyyyMMddHHmmss"

			需与 AccessToken 中时间戳相同
--	--	--	-----------------------

## 2.2 接口安全

- 1) 采用签名的方式
- 2) 流水号做唯一校验

## 2.3 请求方式

采用请求类型为 post 方式 http 请求

## 2.4 消息交换说明

### 2.4.1 表具注册

HTTP 通讯说明:

请求地址: <http://baseUrl/api/v1/collect/meter/1001>

请求类型: POST

请求方向: CIS → DAS

请求/响应:

请求参数				
字段名	变量名	类型	必填	备注
通讯流水	serialNo	String	是	32 位随机唯一流水
用户号	userNo	String	是	唯一标识且长度不超过 17 位
用户名称	username	String	否	
设备 id	deviceId	String	否	
表具厂商	factorNo	String	是	由金卡提供
表具类型	meterType	String	是	具体编码可以参考附录五
表具编码	meterNo	String	是	表号 12 位数字

通讯号	commNo	String	否	如果表有通讯号概念填写，一般表具编码就可
表具底数	initNum	String	是	保留小数 4 位(旧表复用结算时使用)
结算模式	billingModel	String	是	1. 表端结算 2. 中心计费 (CIS 系统) 3、中心计费 (采集系统) 针对 CIS 结算表具，由 CIS 业务系统进行结算并联动阀门开关控制
付费模式	payModel	String	是	1. 预付费，2. 后付费
透支额度	overdraft	String	否	默认 0 (预付费模式有效)
初始周期累积量	initCycle	String	否	换表后希望新表在旧表的某个周期累计量开始周期计算新的累积量
公司编码	companyCode	String	是	由金卡提供燃气公司编码
提交日期	sdateTime	String	是	YYYYMMDDHHMMSS
价格名称	priceCode	String	是	支持中文和编码
地址	address	String	否	设备注册地址
电话	phoneNum	String	否	用户联系电话
用户类别	customerType	String	否	用户类别，详见附录三
<b>响应参数</b>				
响应码	echoCode	String	是	成功是 0，失败是 9999
响应消息	echoMsg	json	是	返回错误信息
以下参数在 echoCode 等于 0000 时有返回				
通讯流水	serialNo	String	是	回送通讯流水
设备 id	deviceId	String	是	唯一的设备 id
响应日期	rdateTime	String	是	YYYYMMDDHHMMSS

## 2.4.2 表具注销

HTTP 通讯说明:

请求地址: <http://baseUrl/api/collect/meter/1002>

请求类型: POST

请求方向: CIS->DAS

请求/响应:

请求参数				
字段名	变量名	类型	必填	备注
通讯流水	serialNo	String	是	32 位随机唯一流水

用户号	userNo	String	是	唯一标识且长度不超过 17 位
设备 id	deviceId	String	否	
表具厂商	factorNo	String	是	由金卡提供
表具编码	meterNo	String	是	表号
表具底数	initNum	String	是	保留小数 4 位
注销原因	reason	String	是	不得超过 100 字符（50 汉字）
公司编码	companyCode	String	是	燃气公司编码
提交日期	sdateTime	String	是	YYYYMMDDHHMMSS
<b>响应参数</b>				
响应码	echoCode	String	是	成功是 0000，失败是 9999
响应消息	echoMsg	json	是	返回错误信息
以下参数在 echoCode 等于 0000 时有返回				
通讯流水	serialNo	String	是	回送通讯流水
设备 id	deviceId	String	是	唯一的设备 id
响应日期	rdateTime	String	是	YYYYMMDDHHMMSS

## 2.4.3 指令下发

HTTP 通讯说明：

请求地址：http://baseUrl/api/collect/meter/1004

请求类型：POST

请求方向：CIS → DAS

请求/响应：

<b>请求参数</b>				
字段名	变量名	类型	必填	备注
通讯流水	serialNo	String	是	32 位随机唯一流水
用户号	userNo	String	是	唯一标识且长度不超过 17 位
设备 id	deviceId	String	否	
表具厂商	factorNo	String	是	金卡提供
表具编码	meterNo	String	是	表号
命令码	paramNo	String	是	见附录一
参数正文	paramContext	json	是	见附录一
公司编码	companyCode	String	是	燃气公司编码
提交日期	sdateTime	String	是	YYYYMMDDHHMMSS
<b>响应参数</b>				
响应码	echoCode	String	是	成功是 0000，失败是 9999
响应消息	echoMsg	json	是	
以下参数在 echoCode 等于 0000 时有返回				



通讯流水	serialNo	String	是	回送通讯流水
设备 id	deviceId	String	是	唯一的设备 id
响应日期	rdateTime	String	是	YYYYMMDDHHMMSS

## 2.4.4 指令取消

HTTP 通讯说明:

请求地址: http://baseUrl/api/collect/meter/1005

请求类型: POST

请求方向: CIS->DAS

请求/响应:

请求参数				
字段名	变量名	类型	必填	备注
通讯流水	serialNo	String	是	32 位随机唯一流水
用户号	userNo	String	是	唯一标识且长度不超过 17 位
设备 id	deviceId	String	否	
表具厂商	factorNo	String	否	金卡提供
表具编码	meterNo	String	是	表号
指令下发流水	sendSerialNo	String	是	1004 指令下发时的通讯流水
公司编码	companyCode	String	是	燃气公司编码
响应参数				
响应码	echoCode	String	是	成功是 0000, 失败是 9999
响应消息	echoMsg	json	是	
以下参数在 echoCode 等于 0000 时有返回				
通讯流水	serialNo	String	是	回送通讯流水
设备 id	deviceId	String	是	唯一的设备 id
响应日期	rdateTime	String	是	YYYYMMDDHHMMSS

## 2.4.5 价格设置(价格新增和价格调价)

HTTP 通讯说明:

请求地址: http://baseUrl/api/collect/meter/1007

请求类型: POST

请求方向: CIS->DAS

请求/响应:

请求参数				
字段名	变量名	类型	必填	备注
通讯流水	serialNo	String	是	32 位随机唯一流水
表具厂商	factorNo	String	是	金卡提供
公司编码	companyCode	String	是	燃气公司编码
价格编码	priceCode	String	是	唯一的价格编码
价格版本	priceVersion	String	是	
价格属性	priceProperty	String	是	1、新增 2、调价
价格生效时间	priceBeginDate	String	否	仅对调价有效，调价必填
周期长度	cycleLength	String	否	单位：月 启用阶梯需填
价格类型	priceType	String	是	1、单价 2、阶梯
清 0 标识	clearFlag	String	否	调价时候要用，0 为清除, 1 为不清除
价格表	priceDetail	json	是	阶梯：{ "priceLength": "2", (价格阶梯数) "price1": "2.5", (1 阶价格) "price2": "3.5", (2 阶价格) "ladder1End": "50" (1 阶止方) } 单价：{ "priceLength": "1", (阶梯长度单价为 1) "price1": "2.5", }
响应参数				
响应码	echoCode	String	是	成功是 0000，失败是 9999
响应消息	echoMsg	String	是	
以下参数在 echoCode 等于 0000 时有返回				
通讯流水	serialNo	String	是	回送通讯流水
设备 id	deviceId	String	是	唯一的设备 id
响应日期	rdateTime	String	是	YYYYMMDDHHMMSS

## 2.4.6 价格修改(换价格)

HTTP 通讯说明:

请求地址：http://baseUrl/api/collect/meter/1008

请求类型：POST

请求方向：CIS→DAS

请求/响应：

请求参数				
字段名	变量名	类型	必填	备注
通讯流水	serialNo	String	是	32 位随机唯一流水
用户号	userNo	String	是	唯一标识且长度不超过 17 位
设备 id	deviceId	String	否	
表具厂商	factorNo	String	是	金卡提供
表具编码	meterNo	String	是	12 位表号
新价格编码	newPriceCode	String	是	
公司编码	companyCode	String	是	燃气公司编码
响应参数				
响应码	echoCode	String	是	成功是 0000，失败是 9999
响应消息	echoMsg	json	是	
以下参数在 echoCode 等于 0000 时有返回				
通讯流水	serialNo	String	是	回送通讯流水
设备 id	deviceId	String	是	唯一的设备 id
响应日期	rdateTime	String	是	YYYYMMDDHHMMSS

## 2.4.7 表具更换

HTTP 通讯说明：

请求地址：http://baseUrl/api/collect/meter/1009

请求类型：POST

请求方向：CIS→DAS

请求/响应：

请求参数				
字段名	变量名	类型	必填	备注
通讯流水	serialNo	String	是	32 位随机唯一流水
用户号	userNo	String	是	唯一标识且长度不超过 17 位
旧表表号	meterNo	String	是	12 位数字

旧表 deviceId	deviceId	String	是	旧表 deviceId 即是注册的设备 id
表具厂商	factorNo	String	是	由金卡统一定，仅支持同一厂家的换表
新表表具类型	newMeterType	String	是	参考附录五
新表表具编码	newMeterNo	String	是	表号
新表通讯号	newCommNo	String	否	如果表有通讯号概念填写，一般表具编码就可
旧表表具读数	oldNum	String	是	保留小数 4 位
新表表具底数	newNum	String	是	保留小数 4 位
公司编码	companyCode	String	是	燃气公司编码
<b>响应参数</b>				
响应码	echoCode	String	是	成功是 0000，失败是 9999
响应消息	echoMsg	json	是	
以下参数在 echoCode 等于 0000 时有返回				
通讯流水	serialNo	String	是	回送通讯流水
新表设备 id	deviceId	String	是	唯一的设备 id
响应日期	rdateTime	String	是	YYYYMMDDHHMMSS

## 2.4.8 批量获取抄表数据(批量获取)

HTTP 通讯说明：

请求地址：http://baseUrl/api/v2/collect/meter/1013

请求类型：POST

请求方向：CIS->DAS

请求/响应：

请求参数				
字段名	变量名	类型	必填	备注
通讯流水	serialNo	String	是	32 位随机唯一流水
公司编码	companyCode	String	是	燃气公司编码
表具厂商	factorNo	String	是	金卡提供
表具编号	meterNo	String	否	不填就是查所有的记录
抄表时间	readingDate	String	否	填写查询填的日期的最近的读数，不填返回最新的日期格式 yyyy-mm-dd
响应参数				
响应码	echoCode	String	是	成功是 0000，失败是 9999
响应消息	echoMsg	json	是	
以下参数在 echoCode 等于 0000 时有返回				

通讯流水	serialNo	String	是	回送通讯流水
抄表数	readingData	Json	是	<pre>{   "total": "54",   "data": [     {       "userArchivesNum": "091203002292",       "valveState": "1",       "gasMeterNo": "201911261531",       "readNum": "0",       "readDate": "2019-11-26 17:27:02",       "meterBalance": "20.6",       "sumGas": "0",       "deviceId": "",       "sumMoney": "0"     }   ] }</pre>
字段名称	数据类型	必填	说明	
total	String	是	总数	
data	Json	是		
userArchivesNum	String	是	用户号	
deviceId	String	是	设备 id	
gasMeterNo	String	是	表号	
readNum	double	是	表最新抄表读数	
meterBalance	double	是	表上最新余额	
readDate	String	是	抄表日期 yyyy-mm-dd hh24:mi:ss	
valveState	String	是	阀门状态 (0:开阀,1:关阀,2:未知)	
sumGas	double	是	累计使用金额	
sumMoney	double	是	累计使用气量	

## 2.4.9 表具详细信息查询

HTTP 通讯说明:

请求地址: <http://baseUrl/api/v2/collect/meter/1014>

请求类型: POST

请求方向: CIS->DAS

## 请求/响应

请求参数				
字段名	变量名	类型	必填	备注
通讯流水	serialNo	String	是	32 位随机唯一流水
用户号	userNo	String	是	不超过 17 位唯一标识
用户名称	userName	String	否	
设备 id	deviceId	String	否	唯一的设备 id
表具厂商	factorNo	String	是	由 CIS 统一定
表具编码	meterNo	String	是	表号
公司编码	companyCode	String	是	
响应参数				
响应码	echoCode	String	是	成功是 0，失败是 9999
响应消息	echoMsg	String	是	返回错误信息
以下参数在 echoCode 等于 0000 时有返回				
通讯流水	serialNo	String	是	回送通讯流水
设备 id	deviceId	String	是	唯一的设备 id
表具编码	meterNo	String	是	表号
表具状态	meterState	String	是	状态 1 :停用 2 启用
安装时间	setUpTime	String	是	yyyy-mm-dd
表底数	beseNum	double	是	
表读数	readingNum	double	是	
用户名称	userName	String	否	
地址	address	String	否	
用户类型	userType	String	否	
阀门状态	valveState	String	是	阀门状态(1:开阀,2:关 3: 未知)
累计使用气量	sumUseGas	double	否	
累计使用金额	sumUseMoney	double	否	
累计充值金额	sumRechargeAmount	double	否	
剩余金额	balance	double	是	
当前价格	price	double	是	
价格名称	priceCode	String	是	
最近抄表时间	lastReadingTime	String	是	yyyy-mm-dd hh24:mi:ss
最后充值金额	lastRechargeAmount	double	是	
最后充值时间	lastRechargeDate	String	是	yyyy-mm-dd hh24:mi:ss
累计充值次数	rechargeCount	String	是	
电池电压值	voltage	String	是	

信号强度	signalStrength	String	是	百分比形式体现
------	----------------	--------	---	---------

## 2.4.10 获取日用量数据(每日结算记录)

HTTP 通讯说明:

请求地址: <http://baseUrl/api/v1/collect/meter/1015>

请求类型: POST

请求方向: CIS->DAS

请求/响应:

请求参数				
字段名	变量名	类型	必填	备注
通讯流水	serialNo	String	是	32 位随机唯一流水
用户号	userNo	String	是	不超过 17 位唯一标识
用户名称	userName	String	否	
设备 id	deviceId	String	否	唯一的设备 id
表具厂商	factorNo	String	是	金卡提供
表具编码	meterNo	String	是	表号
开始时间	startTime	String	是	开始时间 yyyy-mm-dd
结束时间	endTime	String	是	结束时间 yyyy-mm-dd
公司编码	companyCode	String	是	
响应参数				
响应码	echoCode	String	是	成功是 0, 失败是 9999
响应消息	echoMsg	String	是	返回错误信息
以下参数在 echoCode 等于 0000 时有返回				
通讯流水	serialNo	String	是	回送通讯流水
用气数据	gasData	Json	是	以下是 list 字段内容
用气日期	useDate	String	是	yyyy-mm-dd
表具编码	meterNo	String	是	表号
设备 id	deviceId	String	是	唯一的设备 id
用气量	useGas	double	是	
用气金额	useAmount	double	是	
单价	price	double	是	当日跨阶梯有多条

## 2.4.11 表具抄表上报

HTTP 通讯说明:

请求地址: <http://baseUrl/api/collect/meter/2001>

请求类型: POST

请求方向: DAS->CIS

请求/响应:

请求参数				
字段名	变量名	类型	必填	备注
通讯流水	serialNo	String	是	32 位随机唯一流水
用户号	userNo	String	是	不超过 17 位唯一标识
设备 id	deviceId	String	否	
表具厂商	factorNo	String	是	
表具编码	meterNo	String	是	表号
结算模式	billingModel	String	是	1. 表端结算 2. 中心计费
当前抄见数	readNum	String	是	保留小数 4 位
补抄数据	addGas	json	否	如果跟上次上报数据, 中间有数据缺失 (比如多天未通讯), 需要补传数据, 标况读数, 例: { "2018-05-20": "0", "2018-05-21": "0", "2018-05-22": "0", "2018-05-23": "0", "2018-05-24": "0", "2018-05-25": "0", "2018-05-26": "0", "2018-05-27": "0" }
累计使用气量	sumUseGas	String	否	表端结算必填
累计使用金额	sumUseAmount	String	否	表端结算必填
抄见日期	readDate	String	是	YYYYMMDD
抄见时间	readTime	String	是	HHMMSS
电池电压值	batteryCap	String	是	
信号量	semaphore	String	是	显示百分比, 例 98
温度	temperature	String	否	单位为摄氏度
标况总量	standardTotalValue	String	否	标况总量



工况总量	workTotalValue	String	否	工况总量
标况瞬时量	standardInstantValue	String	否	标况瞬时量
工况瞬时量	workInstantValue	String	否	工况瞬时量
压力	pressureValue	String	否	(单位: 千帕)
阀门状态	valveState	String	是	2 阀门打开 3 阀门关闭
表内当前价格	curPrice	String	是	保留小数 4 位 表端结算的表具有此值 中心计费无此值
表内余额	meterBalance	String	否	保留小数 4 位 表端结算的表具有此值 中心计费无此值
公司编码	companyCode	String	是	燃气公司编码
提交日期	sdateTime	String	是	YYYY-MM-DD
响应参数				
响应码	echoCode	String	是	成功是 0000, 失败是 9999
以下参数在 echoCode 等于 0000 时有返回				
通讯流水	serialNo	String	是	回送通讯流水
用户号	userNo	String	是	
设备 id	deviceId	String	是	唯一的设备 id
表具厂商	factorNo	String	是	待定
表具编码	meterNo	String	是	表号
阀门控制	valvecontrol	String	否	0 开阀 1 关阀 2 不处理 3 普通关阀
表当前单价	currentprice	String	否	最新的单价
表当前余量	remainquantity	String	否	金额表是金额, 气量表是气量, 下发单价为 0 时候, 表内的余额 不显示, 只是显示了表读数信息
响应日期	rdateTime	String	是	YYYYMMDDHHMMSS

## 2.4.12 指令下发回调

HTTP 通讯说明:

请求地址: <http://baseUrl/api/collect/meter/2002>

请求类型: POST

请求方向: DAS→CIS

请求/响应:

请求参数				
字段名	变量名	类型	必填	备注
通讯流水	serialNo	String	是	32 位随机唯一流水
用户号	userNo	String	是	不超过 17 位唯一标识
设备 id	deviceId	String	否	
表具厂商	factorNo	String	是	待定
表具编码	meterNo	String	是	表号
指令下发流水	sendSerialNo	String	是	1004 指令下发时的通讯流水
指令回调类型	paramNo	String	是	指令回调类型
执行状态	excuteState	String	是	1 成功 2 失败
公司编码	companyCode	String	是	燃气公司编码
提交日期	sdateTime	String	是	YYYYMMDDHHMMSS
响应参数				
响应码	echoCode	String	是	成功是 0000, 失败是 9999
响应消息	echoMsg	String	是	
以下参数在 echoCode 等于 0000 时有返回				
通讯流水	serialNo	String	是	回送通讯流水
设备 id	deviceId	String	是	唯一的设备 id
响应日期	rdateTime	String	是	YYYYMMDDHHMMSS

### 2.4.13 表具告警上报

HTTP 通讯说明:

请求地址: <http://baseUrl/api/collect/meter/2003>

请求类型: POST

请求方向: DAS → CIS

请求/响应:

请求参数				
字段名	变量名	类型	必填	备注
通讯流水	serialNo	String	是	32 位随机唯一流水
用户号	userNo	String	是	不超过 17 位唯一标识
设备 id	deviceId	String	否	
表具厂商	factorNo	String	是	待定
表具编码	meterNo	String	是	表号
告警分类	alarmType	String	是	见附录二
告警内容	alarmContent	String	是	

告警优先级	alarmPriority	String	是	1: 低, 2: 中, 3: 高
公司编码	companyCode	String	是	燃气公司编码
提交日期	sdateTime	String	是	YYYYMMDDHHMMSS
<b>响应参数</b>				
响应码	echoCode	String	是	成功是 0000, 失败是 9999
响应消息	echoMsg	String	是	
以下参数在 echoCode 等于 0000 时有返回				
通讯流水	serialNo	String	是	回送通讯流水
设备 id	deviceId	String	是	唯一的设备 id
响应日期	rdateTime	String	是	YYYYMMDDHHMMSS

### 3 参考附录

#### 附录一指令参数代码说明

说明	命令码	命令	参数正文模板
设置阀门	03	valveSet	<pre>{   "total": "1",   "data": {     "valveSet": "1"   } }</pre>
字段	数据类型	说明	
total	String	总数	
data	Json		
valveSet	String	(0 为开阀, 1 为关阀)	
充值	05	recharge	<pre>{   "total": "1",   "data": {     "rechargeAmount": "500",     "rechargeCount": "1",     "rechargeDate": "2018-10-19 09:02:41",     "rechargeStream": "12345678"   } }</pre>
字段	数据类型	说明	
Total	String	总是	
data	json		
rechargeAmount	String	充值金额	
rechargeCount	String	充值次数	
rechargeDate	String	充值时间 格式 (yyyy-mm-dd hh24:mi:ss)	
rechargeStream	String	充值流水号	

## 附录二告警信息类型说明

代码	分类	优先级
01	通讯异常	高
02	燃气泄漏	高
03	干簧管坏	高
04	阀门漏气	高
05	阀门异常	高
06	电池故障	高
07	强磁干扰	中

## 附录三用户类型说明

代码	分类
01	工业户
02	居民户
03	壁挂炉户
04	集体户
05	居民采暖户
06	商业户
07	公福户
08	居民煤改气
09	低保户

## 附录四 错误信息说明

代码	说明
----	----

代码	说明
0101	Token 校验未通过(检查 AccessToken 和 Authorization 是否正确)
0103	Token 校验未通过(检查 AccessToken 和 Authorization 是否正确)
0105	Token 校验未通过(检查 AccessToken 和 Authorization 是否正确)
0404	厂商编码(factorNo)错误
9995	服务未启动
9999	参数异常

## 附录五 表具类型说明

表具编码 meterType	说明
441	物联网 10 类型
502	物联网 40 类型
10521	NBIOT10 类型
10522	NBIOT40 类型
10524	金卡热式 R1
10523	温补 10 类型

## 4 接口请求示例

### 4.1 注册请求示例:

120.55.14.102:80/api/v1/collect/meter/1001

http 请求头中配置验证信息如下

Content-Type application/json;charset=utf-8

AccessToken 8C58C258C7ED0AB7574FE4B8B0AA4127 (附录五安全加密算法)

Authorization MDAwMDAwNzI6MjAxOTExMjc5MDAwNTY= (附录五安全加密算法)

request

```
{
  "companyCode": "1000058", --金卡提供
  "priceCode": "20200112", -- 可由价格新建接口新建
  "initNum": "0",
  "username": "测试表号",
  "deviceId": "",
  "userNo": "202001121556", --不超过 17 位唯一标识
  "meterType": "441", ---参考附录五
  "sdateTime": "20200102094101", --YYYYMMDDHHMMSS
  "meterNo": "202001121556", 12 位数字
  "billingModel": "2",
  "factorNo": "0011", --金卡提供
  "payModel": "1",
  "serialNo": "21be5f32e6a0484c9228b262ecc8766a" --32 位流水号
}
```

Response

成功:

```
{
  "echoCode": "0000",
  "echoMsg": {
```

<pre> "serialNo": "40497d1e368d47b5b2d53874e5cccce7c", "deviceId": "201911201801", "rdateTime": "20191123125209" } } </pre>
<pre> { "echoCode": "9999", "echoMsg": "用户号 201911201801 已存在，表具注册失败！" } </pre>

## 4.2 注销请求示例

<p>成功:</p> <p>120.55.14.102:80/api/collect/meter/1002</p> <p>http 请求头中配置验证信息如下</p> <p>Content-Type application/json;charset=utf-8</p> <p>AccessToken 8C58C258C7ED0AB7574FE4B8B0AA4127 (附录五安全加密算法)</p> <p>Authorization MDAwMDAwNzI6MjAxOTExMjcxMDAwNTY= (附录五安全加密算法)</p> <p>Request</p> <pre> {   "serialNo": "23a2352aa088424e8e44b9d95d1ed218",   "userNo": "10010011023",   "deviceId": "123",   "factorNo": "0003",   "meterNo": "202004032112",   "initNum": "0",   "reason": "外部接口表具注销",   "companyCode": "10001253",   "sdateTime": "2019-03-24" } </pre>
<p>Response</p>



成功:

```
{  
  "echoCode": "0000",  
  "echoMsg": {  
    "serialNo": "40497d1e368d47b5b2d53874e5ccce7c",  
    "deviceId": "201911201801",  
    "rdateTime": "20191123125209"  
  }  
}
```

```
{  
  "echoCode": "9999",  
  "echoMsg": "用户号 201911201801 不存在，表具注销失败！" }
```

### 4.3 指令下发示例

成功:

120.55.14.102:80/api/collect/meter/1004

Content-Type application/json;charset=utf-8

AccessToken 8C58C258C7ED0AB7574FE4B8B0AA4127 (附录五安全加密算法)

Authorization MDAwMDAwNzI6MjAxOTExMjcxMDAwNTY= (附录五安全加密算法){

阀控类

```
{  
  "serialNo": "20190211114046",  
  "userNo": "201910281500",  
  "deviceId": "",  
  "factorNo": "0014",  
  "meterNo": "201910281500",  
  "paramNo": "03",
```

```
"paramContext":{
    "total":1,
    "data":{
        "valveSet":1
    }
},
"companyCode":"10001058",
"sdateTime":"20190822103948"
}
```

成功返回

```
{
    "echoCode": "0000",
    "echoMsg": {
        "serialNo": "20190211114046",
        "deviceId": "",
        "rdateTime": "20191217131007"
    }
}
```

异常返回

```
{
    "echoCode": "9999",
    "echoMsg": "用户不存在 "
}
```

## 充值类

```
{
  "serialNo": "20181006085302",
  "userNo": "201911291106",
  "factorNo": "0024",
  "meterNo": "201911291106",
  "paramNo": "05",
  "paramContext": {
    "total": 1,
    "data": {
      "rechargeAmount": "100",
      "rechargeCount": "1",
      "rechargeDate": "2018-11-06 08:09:41",
      "rechargeStream": "12345678"
    }
  },
  "companyCode": "10001058",
  "sdateTime": "201810060853"
}
```

## 成功返回

```
{
  "echoCode": "0000",
  "echoMsg": {
    "serialNo": "20190211114046",
    "deviceId": "",
    "rdateTime": "20191217132230"
  }
}
```

异常返回

```
{  
  "echoCode": "9999",  
  "echoMsg": "用户不存在 "  
}
```

## 4.4 指令取消示例

成功:

```
120.55.14.102:80/api/collect/meter/1005  
Content-Type application/json; charset=utf-8  
AccessToken 8C58C258C7ED0AB7574FE4B8B0AA4127 (附录五安全加密算法)  
Authorization MDAwMDAwNzI6MjAxOTExMjcxMDAwNTY= (附录五安全加密算法){  
  "userNo": "20191106152900",  
  "meterNo": "181929419764",  
  "factorNo": "0017",  
  "sendSerialNo": "219311112",  
  "deviceId": "1bdf3aab3e1a4fa9b3bd46f1f72e9bb6",  
  "serialNo": "e88085b131554af08d17145ffe7d4883"  
}
```

Response

成功返回

```
{
  "echoCode": "0000",
  "echoMsg": {
    "serialNo": "e88085b131554af08d17145ffe7d4883",
    "deviceId": "",
    "rdateTime": "20191217131151"
  }
}
```

异常返回

```
{
  "echoCode": "9999",
  "echoMsg": "用户不存在 "
}
```

## 4.5 价格设置示例

120.55.14.102:80/api/collect/meter/1007

http 请求头中配置验证信息如下

Content-Type application/json;charset=utf-8

AccessToken 8C58C258C7ED0AB7574FE4B8B0AA4127

Authorization MDAwMDAwNzI6MjAxOTExMjcwMDAwNTY=

价格新增请求参数

```
{
  "serialNo": "a25bae8d3cae473dbd3233cdcd1e11111",
  "factorNo": "0025",
  "companyCode": "10001058",
}
```

<pre> "priceCode": "20191218", "priceProperty": "1", "priceBeginDate": "2019-11-23",  "cycleLength": "12", "priceType": "2", "priceDetail": {     "priceLength": "3",     "price1": "1.4",     "ladder1End": "50",     "price2": "2.4",     "ladder2End": "300",     "price3": "3.4" } } </pre>
<p>成功返回</p> <pre> {     "echoCode": "0000",     "echoMsg": {         "serialNo": "a25bae8d3cae473dbd3233cdcd1e111111",         "deviceId": "",         "rdateTime": "20191217132834"     } } </pre> <p>异常返回</p> <pre> { </pre>

```
"echoCode": "2004",  
"echoMsg": "价格信息已存在"  
}
```

#### 调价请求参数

```
{  
  "serialNo": "155634648758500201907031344423580",  
  "factorNo": "0007",  
  "companyCode": "10001059",  
  "priceCode": "201907031439",  
  "priceVersion": "V1.0",  
  "priceProperty": "2",  
  "priceBeginDate": "2019-07-07",  
  
  "cycleLength": "12",  
  "priceType": "2",  
  "priceDetail": {  
    "priceLength": "3",  
    "price1": "1.4",  
    "ladder1End": "50",  
    "price2": "2.4",  
    "ladder2End": "300",  
    "price3": "3.4"  
  },  
  "clearFlag": "0"  
}
```

#### 成功返回

```
{  
  "echoCode": "0000",
```

```
"echoMsg": {
  "serialNo": "a25bae8d3cae473dbd3233cdcd1e11111",
  "deviceId": "",
  "rdateTime": "20191217132834"
}
}
```

异常返回

```
{
  "echoCode": "2004",
  "echoMsg": "价格信息已存在"
}
```

## 4.6 价格修改示例

120.55.14.102:80/api/collect/meter/1008

http 请求头中配置验证信息如下

Content-Type application/json;charset=utf-8

AccessToken 8C58C258C7ED0AB7574FE4B8B0AA4127

Authorization MDAwMDAwNzI6MjAxOTExMjcxMDAwNTY=

```
{
  "serialNo": "075b936e2e9c41d4846794e11b18822d",
  "userNo": "0003",
  "deviceId": "10001058",
  "factorNo": "20190128 新增单价测试 test01",
  "meterNo": "201911291106",
}
```



<pre> "newPriceCode": "2019-01-29" } </pre>
<pre> {   "echoCode": "0000",   "echoMsg": {     "serialNo": "e88085b131554af08d17145ffe7d4883",     "deviceId": "",     "rdateTime": "20191217131151"   } } </pre>

## 4.7 表具更换示例

120.55.14.102:80/api/collect/meter/1009

http 请求头中配置验证信息如下

Content-Type application/json;charset=utf-8

AccessToken 8C58C258C7ED0AB7574FE4B8B0AA4127

Authorization MDAwMDAwNzI6MjAxOTExMjcxMDAwNTY=

```

{
  "serialNo": "b3e4388eaced4441a8d3ce6838e5df3a",
  "userNo": "10001000201903",
  "meterNo": "202004081937",
  "deviceId": "",
  "factorNo": "0003",
  "newMeterType": "502",
  "newMeterNo": "202004081938",
  "newCommNo": "",

```

<pre> "oldNum": "0.0000", "newNum": "0.0000", "companyCode": "1001058" } </pre>
<pre> {   "echoCode": "0000",   "echoMsg": {     "serialNo": "e88085b131554af08d17145ffe7d4883",     "deviceId": "",     "rdatetime": "20191217131151"   } } </pre> <p>异常返回</p> <pre> {   "echoCode": "9999",   "echoMsg": "用户不存在" } </pre>

## 4.8 批量获取抄表数据示例

```

120.55.14.102:80/api/v2/collect/meter/1013
http 请求头中配置验证信息如下
Content-Type application/json;charset=utf-8
AccessToken 8C58C258C7ED0AB7574FE4B8B0AA4127
Authorization MDAwMDAwNzI6MjAxOTExMjcxMDAwNTY=
{
  "companyCode": "10001058",

```

```
"meterNo": "188011110128",  
"factorNo": "0025",  
"readingDate": "",  
"serialNo": "950c892a6b384d3890af48aa69e37a98"  
}}
```

成功返回

```
{  
  "echoCode": "0000",  
  "echoMsg": {  
    "readingData": {  
      "total": "1",  
      "data": [  
        {  
          "sumMoney": "0",  
          "readNum": "2",  
          "userArchivesNum": "188011110128",  
          "valveState": "0",  
          "readDate": "2019-12-23 17:13:24",  
          "meterBalance": "40",  
          "gasMeterNo": "188011110128",  
          "deviceId": "000111201981110002228",  
          "sumGas": "0"  
        }  
      ]  
    },  
    "serialNo": "950c892a6b384d3890af48aa69e37a98"  
  }  
}
```

```
}
```

## 4.9 表具详细信息查询示例

120.55.14.102:80/api/v2/collect/meter/1014

http 请求头中配置验证信息如下

Content-Type application/json;charset=utf-8

AccessToken 8C58C258C7ED0AB7574FE4B8B0AA4127

Authorization MDAwMDAwNzI6MjAxOTExMjcxMDAwNTY=

请求

```
{
  "userNo": "20200218",
  "serialNo": "123",
  "factorNo": "123",
  "deviceId": "",
  "customerName": "",
  "meterNo": "202002201332",
  "companyCode": "10001001"
}
```

返回

```
{
  "echoCode": "0000",
  "echoMsg": {
    "serialNo": "123",
    "meterState": "2",
    "setUpTime": "2020-02-20",
    "beseNum": "1",
    "readingNum": "11",
  }
}
```

```
    "userName": "测试 3",
    "address": "",
    "userType": "居民户",
    "deviceId": null,
    "valveState": "2",
    "sumUseGas": "11",
    "sumUseMoney": "11",
    "priceCode": "民用气价 1 元",
    "sumRechargeAomunt": "222",
    "lastReadingTime": "2020-02-12 00:00:00",
    "balance": "211",
    "meterNo": "202002201332",
    "price": "1",
    "lastRechargeAmount": "1",
    "lastRechargeDate": "2020-02-12 16:06:06"
  }
}
```

## 4.10 获取日用气量数据示例

```
120.55.14.102:80/api/v1/collect/meter/1015
http 请求头中配置验证信息如下
Content-Type application/json;charset=utf-8
AccessToken 8C58C258C7ED0AB7574FE4B8B0AA4127
Authorization MDAwMDAwNzI6MjAxOTExMjcwMDAwNTY=

请求
{
  "companyCode": "10001001",
```

<pre> "userNo": "20200218", "factorNo": "0030", "startTime": "2020-01-01", "endTime": "2020-02-21", "serialNo": "0d2b0793862f4ef8b64649fd3c62f424" } </pre>
<pre> {   "echoCode": "0000",   "echoMsg": {     "serialNo": "0d2b0793862f4ef8b64649fd3c62f424",     "gasData": [       {         "useDate": "2020-01-11",         "price": "1",         "useGas": "2",         "useAmount": "2",         "deviceId": null,         "meterNo": "202002201332"       },       {         "useDate": "2020-01-12",         "price": "1",         "useGas": "4",         "useAmount": "4",         "deviceId": null,         "meterNo": "202002201332"       },       {         "useDate": "2020-02-12", </pre>

```
        "price": "1",
        "useGas": "3",
        "useAmount": "3",
        "deviceId": null,
        "meterNo": "202002201332"
    }
]
}
```

## 4.11 抄表上报示例

```
120.55.14.102:80/api/collect/meter/2001
http 请求头中配置验证信息如下
Content-Type application/json;charset=utf-8
AccessToken 8C58C258C7ED0AB7574FE4B8B0AA4127
Authorization MDAwMDAwNzI6MjAxOTExMjcwMDAwNTY=

{
    "serialNo": "8dc0fb2575ff4f37bccd63288dfb18a2",
    "userNo": "11368453",
    "deviceId": "f522a7db7b164d60aca67c355fa46fe6",
    "factorNo": "0013",
    "meterNo": "181932242531",
    "billingModel": "2",
    "readNum": "0.3",
    "addGas": "",
    "sumUseGas": "0",
    "sumUseAmount": "0",
```

<pre> "readDate": "20200227", "readTime": "111615", "batteryCap": "6.14", "semaphore": "59.0%", "temperature": "", "standardTotalValue": "", "workTotalValue": "", "standardInstantValue": "", "workInstantValue": "", "pressureValue": "", "valveState": "2", "curPrice": "1", "meterBalance": "0", "companyCode": "10001212", "sdateTime": "2020-02-27" } </pre>
<p>成功返回</p> <pre> {   "echoCode": "0000",   "valvecontrol": "2",   "userNo": 182560,   "meterNo": "181932242531",   "remainquantity": 199.301,   "rdateTime": "20200227132436",   "factorNo": "0013",   "currentprice": 2.33,   "serialNo": "8dc0fb2575ff4f37bccd63288dfb18a2" } </pre>



## 4.12 指令下发回掉示例

```
120.55.14.102:80/api/collect/meter/2001
http 请求头中配置验证信息如下
Content-Type application/json;charset=utf-8
AccessToken 8C58C258C7ED0AB7574FE4B8B0AA4127
Authorization MDAwMDAwNzI6MjAxOTExMjc4MDAwNTY=
```

```
{
  "serialNo": "0a06947697db44d7b5e6aa6ce00bac7d",
  "userNo": "11250529",
  "deviceId": "4f404b03c39c4a07b3ead4062d964f9d",
  "factorNo": "0013",
  "meterNo": "191926241615",
  "billingModel": "2",
  "readNum": "0.7",
  "addGas": "",
  "sumUseGas": "0",
  "sumUseAmount": "0",
  "readDate": "20191121",
  "readTime": "12:17:08",
  "batteryCap": "6.36",
  "semaphore": "44.0%",
  "temperature": "",
  "standardTotalValue": "",
  "workTotalValue": "",
  "standardInstantValue": "",
  "workInstantValue": "",
  "pressureValue": "",
```

<pre> "valveState": "2", "curPrice": "0", "meterBalance": "0", "companyCode": "10001212", "sdateTime": "2019-11-21" } </pre>
<p>返回</p> <pre> {   "echoCode": "0000",   "echoMsg": {     "serialNo": "651435288762342361125194147022",     "deviceId": "",     "rdateTime": "20200227132802"   } } </pre>

### 4.13 告警上报示例

<p>120.55.14.102:80/api/collect/meter/2003</p> <p>http 请求头中配置验证信息如下</p> <p>Content-Type application/json;charset=utf-8</p> <p>AccessToken 8C58C258C7ED0AB7574FE4B8B0AA4127</p> <p>Authorization MDAwMDAwNzI6MjAxOTExMjcxMDAwNTY=</p> <pre> {   "serialNo": "0f8a1f70b91b42aa8685b0b3c320c11a19",   "userNo": "090909123123",   "deviceId": "23343239825f44be8d539ec920704d00", </pre>
---

<pre> "factorNo": "0018", "meterNo": "090909123123", "alarmType": "05", "alarmContent": "阀门直通", "alarmPriority": "3", "companyCode": "10001058", "sdateTime": "2019-11-21 16:32:50" } </pre>
<pre> {     "echoCode": "0000",     "echoMsg": "告警上报成功" } </pre>

## 5 安全加密算法

注：appCode 和 appSecret 由金卡系统分配

```

public Encryption(appCode,appSecret) {
    try{
        String time = new SimpleDateFormat("yyyyMMddHHmmss").format(new Date());
        String AccessToken = encodeMd5(appCode + appSecret + time);
        String Authorization = encodeString(appCode + ":" + time);

        System.out.println("AccessToken:"+AccessToken+"----"+"Authorization:"+Authorization);
    }catch (Exception e) {
        e.printStackTrace();
    }
}

public String encodeMd5(String str){
    MessageDigest md;
    try {
        md = MessageDigest.getInstance("MD5");
    } catch (NoSuchAlgorithmException e) {
        throw new RuntimeException(e.getCause()+e.getMessage());
    }
}

```

---

```
byte[] result=md.digest(str.getBytes(Charset.forName("UTF-8")));

StringBuffer sb=new StringBuffer();
for(int i=0;i<result.length;i++){
    String tmp=StringUtils.leftPad(String.format("%X", result[i]), 2,'0');
    sb.append(tmp);
}
return sb.toString();
}
```

```
public String encodeString(String str) throws IOException {
    sun.misc.BASE64Encoder encoder = new sun.misc.BASE64Encoder();
    String encodedStr = new String(encoder.encodeBuffer(str.getBytes()));
    return encodedStr.trim();
}
```