**说明**

本接口采用同步车位停车状态的方法，针对获取车辆进出信息的需求。

HTTP方式推送，BODY为JSON结构

接收方可以验证签名sign，确认每笔连接的有效性。

发送方在推送程序启动的时，将所有车位的最新数据推送一遍，之后只在车辆进出时实时增量发送，收不到应答，会重复补发。

签名：code,key,user,pass,stamp按字段名排序后，对其值连接后，采用MD5加密，stamp为随机数据，例子如下

TreeMap<String, String> map = **new** TreeMap<String, String>();

map.put("code", code);

map.put("key", key);

map.put("user", user);

map.put("pass", pass);

map.put("stamp", stamp);

StringBuffer sb = new StringBuffer();

**for**(String value : map.values()){

sb.append(value);

}

**String sign =** *Encrypt*(sb.toString());

**private** **static** String Encrypt(String strSrc) **throws** Exception {

**byte**[] bt = strSrc.getBytes();

MessageDigest md = MessageDigest.*getInstance*("MD5");

md.update(bt);

String strDes = *bytes2Hex*(md.digest());

**return** strDes;

}

**public** **static** String bytes2Hex(**byte**[] bts) {

String des = "";

**for** (**int** i=0; i<bts.length; i++) {

String tmp = (Integer.*toHexString*(bts[i] & 0xFF));

**if**(tmp.length()==1) des += "0";

des += tmp;

}

**return** des;

}

**List Of API**

|  |  |  |
| --- | --- | --- |
| # | Functional Description | Remarks |
| 1 | 同步停车状态 |  |

**API#1:同步停车状态**

The data pushed is a JSON object, its properties are described as follows:

|  |  |  |  |
| --- | --- | --- | --- |
| Property name | Data type | Description | Remarks |
| code | String | 功能码 | 值32 |
| key | int | 包序号 | 接收方原样返回 |
| sign | String | 签名 |  |
| data | array | 停车数据 | Each array element is an object, the  properties are as follows:  •r(string)停车场号，可按接收方要求定义，可缺  •i(string)车位号,可按接收方要求定义,必须  •m(string)设备IMEI号，可缺  •c(string)是否停车,必须,Y停车,N无车  •t(long) 1970.1.1 UTC 零点开始到现在的时间毫秒数，可缺  •v(string)电压,如3.57，可缺  •s(string)信号强度,如-85.7，可缺  •z(string)信噪比,如10.8，可缺  •o(string)是否在线,如Y/N，可缺  数组一般一次最多20-50行数据 |

It is recommended that the return data is a JSON object and the following properties are included:

|  |  |  |  |
| --- | --- | --- | --- |
| Property name | Data type | Description | Remarks |
| code | string | 功能码 | 与请求一致 |
| key | Int | 包序号 | 与请求一致 |
| err | string | 错误码 | 0正确，收到数据即认为推送通信正常，数据结构正常，接收方应该返回正确码。如果有数据错误，接收方应整包不处理，返回错误。 |

Example:

REQUEST：

{

“code”:”32”,

“key”:34,

“sign”: “skkdkk3200skwkekfeeepppawfwiiqwi9909jgej”,

“data”:[{ “r”, “1002” ,

“i”: “P6-5234”,

“c”: “Y” ,

“m”: “869405034679341” ,

“t”: 1470000000000,

“v”: “3.54” ,

“s”: “-74.5” ,

“z”: “10.6” ,

“o”: “Y”

}]

}

RESPONSE:

{

“code”:”32”,

“key”: 34,

“err”: “0”

}